EX.NO.:1(a)	SOLVE PROBLEMS BY USING SEQUENTIAL SEARCH
DATE:	

AIM:

To develop a Java application to search an element in an array using sequential search algorithm.

ALGORITHM:

Step 1: Start the program

Step 2: Declare an array and search element as key.

Step 3: Traverse the array until the key is found.

Step 4: If the key is found, return the index position of the array element

Step 5: If the key element is not found, return -1.

Step 6: Stop the program.

PROGRAM:

LinearSearch.java

```
public class LinearSearch
{
	static intsearch(intarr[], int n, int s)
	{
		for (inti = 0; i< n; i++)
		{
			if (arr[i] == s)
				return i;
		}
		return -1;
	}
	public static void main(String[] args)
	{
		int[] arr = { 3, 4, 1, 7, 5 };
		int n = arr.length;
		int s = 4;
		int index = search(arr, n, s);
		if (index == -1)
			System.out.println("Element is not present in the array");
		else
			System.out.println("Element found at index " + index);
	}
}
```

OUTPUT:

```
D:\Java\CS3381>javac LinearSearch.java
D:\Java\CS3381>java  LinearSearch
Element found at index 1
```

RESULT:

Thus, the Java application to perform sequential search was implemented and executed successfully.

EX.NO.:1(b)	SOLVE PROBLEMS BY USING BINARY SEARCH
DATE:	

AIM:

To develop a Java application to search an element in an array using binary search algorithm.

ALGORITHM:

Step 1: Start the program.

Step 2: Declare an array and search element as key**.**

Step 3: Compare search key with the middle element.

Step 4: If key matches with the middle element,return the mid index.

Step 5: Otherwise, if key is greater than the mid element recur for the right half.

Step 6: Otherwise (s is smaller) recur for the left half.

Step 7: Stop the program.

PROGRAM:

BinarySearch.java

```
class BinarySearch
{
 public static intbinarySearch(intarr[], int first, int last, int key)
        {
                if (last>=first)
                {
                        int mid = (first +last)/2;
                        if (arr[mid] == key)
                        {
                                return mid;
                        }
                        else if (arr[mid] >key)
                        {
                                return binarySearch(arr, first, mid-1, key);
                        }
                        else
                        {
                                return binarySearch(arr, mid+1, last, key);
                        }
                }
```

```
                return -1;
        }
        public static void main(String args[])
        {
                intarr[] = {10,20,30,40,50};
                int key = 30;
                int last=arr.length-1;
                int result = binarySearch(arr,0,last,key);
                if (result == -1)
                        System.out.println("Element is not found!");
                else
                        System.out.println("Element is found at index: "+result);
        }
}
```

OUTPUT:

```
D:\Java\CS3381>javac BinarySearch.java
D:\Java\CS3381>java  BinarySearch
Element is found at index: 2
D:\Java\CS3381>
```

RESULT:

Thus, the Java application to perform binary search was implemented and executed successfully.

EX.NO.:1(c)	SOLVE PROBLEMS BY USING QUADRATIC SORTING ALGORITHMS- SELECTION SORT
DATE:	

AIM:

To develop a Java application to sort an array of elements in ascending order using selection sort.

ALGORITHM:

Step 1: Start the program.

Step 2: Select the first unsorted element as the minimum.

Step 3: For each of the unsorted elements, if the element is <minimum, set element as new minimum.

Step 4: Swap minimum with first unsorted position.

Step 5: Repeat steps 2-4 for (n-1) elements until the list is sorted.

Step 6: Print the sorted array.

Step 7: Stop the program.

PROGRAM:

SelectionSort.java

```
public class SelectionSort
{
        public static void selectionsort(int[] arr)
        {
                int n=arr.length;
                for(inti=0;i<n-1;i++)
                {
                        intmin=i;
                        for(int j=i+1;j<n;j++)
                        {
                                if(arr[j]<arr[min])
                                {
                                        min=j;
                                }
                        }
```

```
                int temp=arr[i];
                arr[i]=arr[min];
                arr[min]=temp;
            }
        }
        public static void main(String[] args)
        {
            int[] arr= {15,21,6,3,19,20};
            System.out.println("Elements in the array before Sorting");
            for(int i:arr)
                System.out.print(i+" ");
            selectionsort(arr);
            System.out.println("\nElements in the array after Sorting");
            for(int i:arr)
                System.out.print(i+" ");
        }
}
```

OUTPUT:

```
D:\Java\CS3381>javac SelectionSort.java
D:\Java\CS3381>java  SelectionSort
Elements in the array before Sorting
15 21 6 3 19 20
Elements in the array after Sorting
3 6 15 19 20 21
D:\Java\CS3381>
```

RESULT:

Thus, the Java application to sort an array of N elements using selection sort was implemented and executed successfully.

EX.NO.:1(d)	SOLVE PROBLEMS BY USING QUADRATIC SORTING ALGORITHMS-INSERTION SORT
DATE:	

AIM:

To develop a Java application to sort an array of elements in ascending order using insertion sort.

ALGORITHM:

Step 1: Start the program.

Step 2: Define an array num to store N numbers for insertion sort**.**

Step 3: Run an outer loop i from 1 to N to repeat the process.

Step 4: Store the number num[i] to be inserted at proper place in variable x.

Step 5: Run a while loop j inside the body of the outer loop i from i-1 to 0.

Step 6: Check if the value of x is less than value of num[j] then shift the number num[j] towards right else break the inner loop j.

Step 7: Outside the body of inner loop j insert the value of x at num[j+1] position.

Step 8: Print the sorted array.

Step 9: Stop the program.

PROGRAM:

InsertionSort.java

```
public class InsertionSort
{
	public static void main(String args[])
	{
		intnum[]= {12,9,37,86,2,17,5};
		inti,j,x;
		System.out.println("Array before Insertion Sort");
		for(i=0; i<num.length; i++)
		{
			System.out.print(num[i]+" ");
		}
		for(i=1; i<num.length; i++)
		{
		x=num[i];
		j=i-1;
		while(j>=0)
		{
			if(x<num[j])
			{
```

```
                                num[j+1]=num[j];
                        }
                        else
                        {
                                break;
                        }
                        j=j-1;
                         }
                        num[j+1]=x;
                }
                System.out.print("\n\nArray after Insertion Sort\n");
                for(i=0; i<num.length; i++)
                {
                        System.out.print(num[i]+" ");
                }
        }
}
```

OUTPUT:

```
D:\Java\CS3381>javac InsertionSort.java

D:\Java\CS3381>java  InsertionSort

Array before Insertion Sort

12 9 37 86 2 17 5

Array after Insertion Sort

2 5 9 12 17 37 86

D:\Java\CS3381>
```

RESULT:

Thus, the Java application to sort an array of N elements using insertion sort was implemented and executed successfully.

EX.NO.: 2(a)	DEVELOP STACK DATA STRUCTURES USING CLASSES AND OBJECTS
DATE:	

AIM:

To develop a Java program to implement stack data structure using classes and objects.

ALGORITHM:

Step 1: Start the program.

Step 2: Create a class named Stack and declare the instance variables st[], top, maxsize as private.

Step3: Use constructor to allocate memory space for the array and initialize top as -1.

Step 4: Define methods such as isFull(), isEmpty(), push(), pop() and printStack();

Step 5: Push Operation

Step 5.1: Check whetherstack has some space or stack is full.

Step 5.2: If the stack has no space then display "overflow" and exit.

Step 5.3: If the stack has space then increase top by 1 to point next empty space.

Step 5.4: Add element to the new stack location, where top is pointing.

Step 5.5: Push operation performed successfully.

Step 6: Pop operation

Step 6.1: Check whether stack has some element or stack is empty.

Step 6.2: If the stack has no element means it is empty then display "underflow"

Step 6.3: If the stack has some element, accesses the data element at which top is pointing.

Step 6.4: Decrease the value of top by 1.

Step 6.5: Pop operation performed successfully.

Step 7: PrintStack operation

Step 7.1:Check whether stack has some element or stack is empty.

Step 7.2: If the stack has no element means it is empty then display "underflow".

Step 7.3: If the stack has some element, traverse the array from top to bottom and display the elements.

Step 8: Define the main() method

Step 9: Create object of Stack class.

Step 10: Display the menu and get the user choice and invoke appropriate method.

Step 11: Stop the program.

PROGRAM:

Stack.java

```
import java.util.Scanner;
public class Stack
{
        private intmaxsize, top;
        private int[] st;
        public Stack(int size)
        {
                maxsize = size;
                st = new int[maxsize];
                top = -1;
        }
        booleanisEmpty()
        {
                return top==-1;
        }
        booleanisFull()
        {
                return top==maxsize-1;
        }
        public void push(int element)
        {
                 if(isFull())
                        System.out.println("Overflow");
                else
                        st[++top] = element;
        }
        public intpop()
        {
                 if(isEmpty())
                {
                        System.out.println("UnderFlow");
                return (-1);
                }
                 return (st[top--]);
        }
        public void printStack()
        {
                System.out.println("Stack Elements:");
                for (int i = top; i>=0; i--)
```

```
                System.out.println(st[i]);
        }
        public static void main(String[] args)
        {
                Scanner sc = new Scanner(System.in);
                System.out.println("Enter stack size");
                int size=sc.nextInt();
                Stack obj = new Stack(size);
                while (true)
                {
                        System.out.println("\nSTACK\n*****\n1.PUSH\n2.POP
                        \n3.Display\n4.EXIT\nEnter your choice");
                        intch = sc.nextInt();
                switch (ch)
                {
                        case 1:
                                        System.out.println("Enter Element");
                                int n = sc.nextInt();
                                        obj.push(n);
                                break;
                        case 2:
                                        System.out.printf("Poped element is %d", obj.pop());
                                break;
                        case 3:
                                        obj.printStack();
                                break;
                        case 4:
                                        System.exit(0);
                        default:
                                        System.out.println("Wrong option");
                }
                }
        }
}
```

OUTPUT:

```
D:\Java\CS3381>java  Stack
Enter stack size
5

STACK
*****
1.PUSH
2.POP
```

```
3.Display
4.EXIT
Enter your choice
1
Enter Element
12

STACK
*****
1.PUSH
2.POP
3.Display
4.EXIT
Enter your choice
1
Enter Element
34

STACK
*****
1.PUSH
2.POP
3.Display
4.EXIT
Enter your choice
1
Enter Element
56

STACK
*****
1.PUSH
2.POP
3.Display
4.EXIT
Enter your choice
1
Enter Element
78

STACK
*****
1.PUSH
2.POP
3.Display
4.EXIT
Enter your choice
3
Stack Elements:
78
```

```
56
34
12

STACK
*****
1.PUSH
2.POP
3.Display
4.EXIT
Enter your choice
2
Poped element is 78
STACK
*****
1.PUSH
2.POP
3.Display
4.EXIT
Enter your choice
3
Stack Elements:
56
34
12

STACK
*****
1.PUSH
2.POP
3.Display
4.EXIT
Enter your choice
4

D:\Java\CS3381>
```

RESULT:

Thus, the Java program to implement stack data structure using classes and objects has developed and executed successfully.

EX.NO.: 2(b)	DEVELOP QUEUE DATA STRUCTURES USING CLASSES AND OBJECTS
DATE:	

AIM:

To develop a Java program to implement queue data structure using classes and objects.

ALGORITHM:

Step 1: Start the program.

Step 2: Create a class named Queue and declare the instance variables items[], front, rear, maxsize as private.

Step3: Use constructor to allocate memory space for the array and initialize front as -1 and rear as -1.

Step 4: Define methods such as isFull(), isEmpty(), enQueue(), deQueue() and display();

Step 5: Enqueue Operation

Step 5.1: Check whetherqueue has some space or queue is full.

Step 5.2: If the queue has no space then display "overflow" and exit.

Step 5.3: If the queue has space then increase rear by 1 to point next empty space.

Step 5.4: Add element to the new location, where rear is pointing.

Step 5.5: Enqueue operation performed successfully.

Step 6: Dequeue operation

Step 6.1: Check whether queue has some element or queue is empty.

Step 6.2: If the queue has no element means it is empty then display "underflow"

Step 6.3: If the queue has some element, accesses the data element at which front is pointing.

Step 6.4: Incrementthe value of front by 1.

Step 6.5: Dequeue operation performed successfully.

Step 7: Displayoperation

Step 7.1: Check whether queue has some element or queue is empty.

Step 7.2: If the stack has no element means it is empty then display "underflow".

Step 7.3: If the stack has some element, traverse the array from front to rear and display the elements.

Step 8: Define the main() method

Step 9: Create object of Queue class.

Step 10: Display the menu and get the user choice and invoke appropriate method.

Step 11: Stop the program.

PROGRAM:
Queue.java

```
import java.util.Scanner;
public class Queue
{
        private intitems[];
        private intmaxsize, front, rear;
        Queue(int size)
        {
                maxsize=size;
                items = new int[size];
                front = -1;
                rear = -1;
        }
        booleanisFull()
        {
                if (front == 0 && rear ==maxsize-1)
                {
                        return true;
                }
                return false;
        }
        booleanisEmpty()
        {
                if (front == -1)
                        return true;
                else
                        return false;
        }
        void enQueue(int element)
        {
                if (isFull())
                {
                        System.out.println("Queue is full");
```

```
        }
        else
        {
                if (front == -1)
                        front = 0;
                rear++;
                items[rear] = element;
                System.out.println("Inserted " + element);
        }
}
intdeQueue()
{
        int element;
        if (isEmpty())
        {
                System.out.println("Queue is empty");
                return (-1);
        }
        else
        {
                element = items[front];
                if (front >= rear)
                {
                        front = -1;
                        rear = -1;
                }
                else
                {
                        front++;
                }
                return (element);
        }
}
void display()
```

```
{
        inti;
        if (isEmpty())
        {
                System.out.println("Empty Queue");
        }
        else
        {
                System.out.println("\nFront index-> " + front);
                System.out.println("Items -> ");
                for (i = front; i<= rear; i++)
                        System.out.print(items[i] + "  ");
                System.out.println("\nRear index-> " + rear);
        }
}
public static void main(String[] args)
{
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter queue size");
        int size=sc.nextInt();
        Queue obj = new Queue(size);
        while (true)
        {
                System.out.println("\nQUEUE\n*****\n1.ENQUEUE\n2.DEQUEUE\
                n3.DISPLAY\n4.EXIT\nEnter your choice");
                intch = sc.nextInt();
                switch (ch)
                {
                        case 1:
                                System.out.println("Enter Element");
                                int n = sc.nextInt();
                                obj.enQueue(n);
                                break;
                        case 2:
```

```
                    System.out.printf("Dequeued element is %d", obj.deQueue());
                    break;
                case 3:
                    obj.display();
                    break;
                case 4:
                    System.exit(0);
                default:
                    System.out.println("Wrong option");
            }
        }
    }
}
```

OUTPUT:

```
D:\Java\CS3381>javac Queue.java

D:\Java\CS3381>java  Queue
Enter queue size
5

QUEUE
*****
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter your choice
1
Enter Element
12
Inserted 12

QUEUE
*****
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter your choice
1
Enter Element
34
Inserted 34
```

```
QUEUE
*****
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter your choice
1
Enter Element
56
Inserted 56

QUEUE
*****
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter your choice
1
Enter Element
78
Inserted 78

QUEUE
*****
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter your choice
3

Front index-> 0
Items ->
12  34  56  78
Rear index-> 3

QUEUE
*****
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter your choice
2
Poped element is 12
QUEUE
*****
1.ENQUEUE
```

```
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter your choice
3

Front index-> 1
Items ->
34  56  78
Rear index-> 3

QUEUE
*****
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter your choice
4

D:\Java\CS3381>
```

RESULT:

Thus, the Java program to implement queue data structure using classes and objects has developed and executed successfully.

EX.NO.: 3	**Develop a java application with an Employee class with Emp_name, Emp_id, Address, Mail_id, Mobile_no as members. Inherit the classes, Programmer, Assistant Professor, Associate Professor and Professor from employee class. Add Basic Pay (BP) as the member of all the inherited classes with 97% of BP as DA, 10 % of BP as HRA, 12% of BP as PF, 0.1% of BP for staff club funds. Generate pay slips for the employees with their gross and net salary**
DATE:	

AIM:

To develop a java application to generate pay slip for different category of employees using the concept of inheritance.

ALGORITHM:

Step 1: Start the program.

Step 2: Create the class **Employee** with name, Empid, address, mailid, mobileno as fields.

Step 3: Inherit the classes **Programmer**, **AssistantProfessor**, **AssociateProfessor** and **Professor** from employeeclass.

Step 4: Add Basic Pay (BP) as the member of all the inherited classes.

Step 5: Calculate DA as 97% of BP, HRA as 10% of BP, PF as 12% of BP, Staff club fund as 0.1% of BP.

Step 6: Calculate gross salary and net salary.

Grosssal=BP+HRA+DA

NetSal=GrossSal-(PF+Staff Club Fund)

Step 7: Create a test class **PaySlip.** Reademployee details, choice and basic pay from user.

Step 8: Based on user choice and input create the object and invoke the necessary methods to display the Payslip.

Step 9: Stop the program.

PROGRAM:

PaySlip.java

```
import java.util.Scanner;
class Employee
{
   String Emp_name,Mail_id,Address,Emp_id, Mobile_no;
   double BP,GP,NP,DA,HRA,PF,CF;
   Scanner get = new Scanner(System.in);
Employee()
   {
```

```
        System.out.println("Enter Name of the Employee:");
        Emp_name = get.nextLine();
        System.out.println("Enter Address of the Employee:");
        Address = get.nextLine();
        System.out.println("Enter ID of the Employee:");
        Emp_id = get.nextLine();
        System.out.println("Enter Mobile Number:");
        Mobile_no = get.nextLine();
        System.out.println("Enter E-Mail ID of the Employee :");
        Mail_id = get.nextLine();
    }
    void display()
    {
        System.out.println("Employee Name: "+Emp_name);
        System.out.println("Employee Address: "+Address);
        System.out.println("Employee ID: "+Emp_id);
        System.out.println("Employee Mobile Number: "+Mobile_no);
        System.out.println("Employee E-Mail ID"+Mail_id);
         DA=BP*0.97;
         HRA=BP*0.10;
         PF=BP*0.12;
         CF=BP*0.01;
         GP=BP+DA+HRA;
         NP=GP-PF-CF;
        System.out.println("Basic Pay :"+BP);
        System.out.println("Dearness Allowance : "+DA);
        System.out.println("House Rent Allowance :"+HRA);
        System.out.println("Provident Fund :"+PF);
        System.out.println("Club Fund :"+CF);
        System.out.println("Gross Pay :"+GP);
        System.out.println("Net Pay :"+NP);
    }
}
class Programmer extends Employee
{
        Programmer()
        {
                System.out.println("Enter Basic pay of the Programmer:");
                BP = get.nextFloat();
        }
        void display()
        {
        System.out.println("==============================="+"\n"+"Programmar
        Pay Slip"+"\n"+"==============================="+"\n");
        super.display();
        }
}
class AssistantProfessor extends Employee
{
        AssistantProfessor()
```

```
        {
                System.out.println("Enter Basic pay of the Assistant Professor:");
                BP = get.nextFloat();
         }
        void display()
        {
      System.out.println("==============================="+"\n"+"Assistant ProfessorPay
Slip"+"\n"+"==============================="+"\n");
super.display();
   }
}
class AssociateProfessor extends Employee
{
        AssociateProfessor()
        {
                System.out.println("Enter Basic pay of the Professor:");
                BP = get.nextFloat();
        }
        void display()
        {
         System.out.println("==============================="+"\n"+"Associate
        Professor Pay Slip"+"\n"+"==============================="+"\n");
        super.display();
        }
}
class Professor extends Employee
{
        Professor()
        {
                System.out.println("Enter Basic pay of the Professor:");
                BP = get.nextFloat();
         }
        void display()
        {
        System.out.println("==============================="+"\n"+"Professor Pay
        Slip"+"\n"+"==============================="+"\n");
        super.display();
        }
}
class Payslip
{
   public static void main(String[] args)
   {
        char ans;
        Scanner sc = new Scanner(System.in);
        do
        {
                System.out.println("Main Menu");
                System.out.println("1. Programmer \n2. Assistant Professor \n3. Associate
                Professor \n4. Professor");
```

```
                System.out.println("Enter your choice: ");
                int choice=sc.nextInt();
                switch(choice)
                {
                                case 1:
                                        Programmer p=new Programmer();
                                        p.display();
                                        break;
                                case 2:
                                        AssistantProfessorap=new AssistantProfessor();
                                        ap.display();
                                        break;
                                case 3:
                                        AssociateProfessor asp=new AssociateProfessor();
                                        asp.display();
                                        break;
                                case 4:
                                        Professor PR=new Professor();
                                        PR.display();
                                        break;
                }
                System.out.println("Do you want to go to Main Menu?(y/n): ");
                ans=sc.next().charAt(0);
                }while(ans=='y'||ans=='Y');
        sc.close();
   }
}
```

OUTPUT:

```
Main Menu
1. Programmer
2. Assistant Professor
3. Associate Professor
4. Professor
Enter your choice: 1
Enter Name of the Employee: Jeremiah
Enter Address of the Employee: Chennai
Enter ID of the Employee: 12345
Enter Mobile Number: 9360362173
Enter E-Mail ID of the Employee : jeremiahe@gmail.com
Enter Basic pay of the Programmer: 56000
==============================
Programmar Pay Slip
==============================
Employee Name: Jeremiah
Employee Address: Chennai
```

Employee ID: 12345
Employee Mobile Number: 9360362173
Employee E-Mail ID : jeremiahe@gmail.com
Basic Pay :56000.0
Dearness Allowance : 54320.0
House Rent Allowance :5600.0
Provident Fund :6720.0
Club Fund :560.0
Gross Pay :115920.0
Net Pay :108640.0
Do you want to go to Main Menu?(y/n): n

RESULT:

Thus, the Java application to generate pay slip for different category of employees was implemented using inheritance and the program was executed successfully.

EX.NO.:4	**Write a Java Program to create an abstract class named Shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contains only the method printArea( ) that prints the area of the given shape**
DATE:	

AIM:

To write a Java program to calculate the area of rectangle, circle and triangle using the concept of abstract class.

ALGORITHM:

Step 1: Start the program.

Step 2: Create an abstract class named shape that contains two integers and an empty method named printArea().

Step 3: Create three classes named rectangle, triangle and circle such that each one of the classes extends the class Shape.

Step 4: Each of the inherited class from shape class should provide the implementation for the methodprintArea().

Step 5: In printAree() method get the input from user and calculate the area of rectangle, circle and triangle.

Step 6: In the AbstractArea, create the objects for the three inherited classes and invoke the methods and display the area values of the different shapes.

Step 7: Stop the program.

PROGRAM:

AbstractArea.java

```
import java.util.*;
abstract class Shape
{
        inta,b;
        abstract void printArea();
}
class Rectangle extends Shape
{
        void printArea()
        {
                System.out.println("\t\tCalculating Area of Rectangle");
                Scanner input=new Scanner(System.in);
                System.out.print("Enter length: ");
                a=input.nextInt();
                System.out.print("\nEnter breadth: ");
```

```
                b=input.nextInt();
                int area=a*b;
                System.out.println("Area of Rectangle: "+area);
        }
}
class Triangle extends Shape
{
        void printArea()
        {
                System.out.println("\t\tCalculating Area of Triangle");
                Scanner input=new Scanner(System.in);
                System.out.print("Enter height: ");
                a=input.nextInt();
                System.out.println("\nEnter breadth: ");
                b=input.nextInt();
                double area=0.5*a*b;
                System.out.println("Area of Triangle: "+area);
        }
}
class Circle extends Shape
{
        void printArea()
        {
                System.out.println("\t\tCalculating Area of Circle");
                Scanner input=new Scanner(System.in);
                System.out.print("Enter radius: ");
                a=input.nextInt();
                double area=3.14*a*a;
                System.out.println("Area of Circle: "+area);
        }
}
class AbstractArea
{
        public static void main(String[] args)
        {
                Shape obj;
                obj=new Rectangle();
                obj.printArea();
                obj=new Triangle();
                obj.printArea();
                obj=new Circle();
                obj.printArea();
        }
}
```

OUTPUT:

```
                Calculating Area of Rectangle
Enter length: 10
Enter breadth: 20
Area of Rectangle: 200
                Calculating Area of Triangle
Enter height: 34

Enter breadth: 56
Area of Triangle: 952.0
                Calculating Area of Circle
Enter radius: 23
Area of Circle: 1661.06
```

RESULT:

Thus, the Java program to calculate the area of rectangle, circle and triangle using the concept of abstract class wasdeveloped and executed successfully.

EX.NO.: 5	**Write a Java Program to create an abstract class named Shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contains only the method printArea( ) that prints the area of the given shape. Solve the above problem using an interface**
DATE:	

AIM:

To write a Java program to calculate the area of rectangle, circle and triangle by implementing the interface shape.

ALGORITHM:

Step 1: Start the program.

Step 2: Create an interface named shape that contains an empty methodnamed printArea().

Step 3: Create three classes named rectangle, triangle and circle such that each one of the classes implements the class Shape.

Step 4: Each of the class should provide the implementation for the method printArea().

Step 5: In printAree() method get the input from user and calculate the area of rectangle, circle and triangle.

Step 6: In the TestAreaclass, create the objects for the three classes and invoke the MethodprintArea() and display the area values of the different shapes.

Step 7: Stop the program.

PROGRAM:

TestArea.java

```
import java.util.Scanner;
interface Shape
{
	public void printArea();
}
class Rectangle implements Shape
{
	public void printArea()
	{
		System.out.println("\t\tCalculating Area of Rectangle");
		Scanner input=new Scanner(System.in);
		System.out.print("Enter length: ");
		int a=input.nextInt();
		System.out.print("\nEnter breadth: ");
		int b=input.nextInt();
		int area=a*b;
		System.out.println("Area of Rectangle: "+area);
```

```
        }
}
class Triangle implements Shape
{
        public void printArea()
        {
                System.out.println("\t\tCalculating Area of Triangle");
                Scanner input=new Scanner(System.in);
                System.out.print("Enter height: ");
                int a=input.nextInt();
                System.out.print("\nEnter breadth: ");
                int b=input.nextInt();
                double area=0.5*a*b;
                System.out.println("Area of Triangle: "+area);
        }
}
class Circle implements Shape
{
        public void printArea()
        {
                System.out.println("\t\tCalculating Area of Circle");
                Scanner input=new Scanner(System.in);
                System.out.print("Enter radius: ");
                int a=input.nextInt();
                double area=3.14*a*a;
                System.out.println("Area of Circle: "+area);
        }
}
public class TestArea
{
        public static void main(String[] args)
        {
                Shape obj;
                obj=new Rectangle();
                obj.printArea();
                obj=new Triangle();
                obj.printArea();
                obj=new Circle();
                obj.printArea();
        }
}
```

OUTPUT:

```
D:\Java\CS3381>javac TestArea.java

D:\Java\CS3381>java  TestArea
        Calculating Area of Rectangle
Enter length: 14

Enter breadth: 24
Area of Rectangle: 336
        Calculating Area of Triangle
Enter height: 45

Enter breadth: 32
Area of Triangle: 720.0
        Calculating Area of Circle
Enter radius: 5
Area of Circle: 78.5

D:\Java\CS3381>
```

RESULT:

Thus, the Java program to calculate the area of rectangle, circle and triangle using the concept of interfacewasdeveloped and executed successfully.

EX.NO.:6	IMPLEMENT EXCEPTION HANDLING AND CREATION OF USER DEFINED EXCEPTIONS
DATE:	

AIM:

To write a Java program to implement user defined exception handling.

ALGORITHM:

Step 1: Start the program.

Step 2: Create a class NegativeAmtException which extends Exception class.

Step 3: Create a constructor which receives the string as argument.

Step 4: Create a class named BankAccount with a constructor and methods such as deposit(), withdraw(), getBalance() and toString().

Step 5: Inside the constructor, deposit() and withdraw() methods check for negative amount. If the amount is negative, then exception will be generated and thrown. Make these methods to specify that NegativeAmtException will be thrown.

Step 6: Create a test class UserDefinedException . Read the account number and initial balance from user and create object of BankAccount class.

Step 7: Display the menu and get the user choice and execute the required operation. Write the code that generates exception, in try block.

Step 8: Use catch block to catch and handle NegativeAmtException. Display the caught exception.

Step 9: Stop the program.

PROGRAM:

UserDefinedException.java

```
import java.util.*;
class NegativeAmtException extends Exception
{
        String msg;
        NegativeAmtException(String msg)
        {
                this.msg=msg;
        }
        public String toString()
        {
                return msg;
        }
}
```

```
class BankAccount
{
	private double balance;
	private intaccountNumber;
	public BankAccount(intaccountNumber,doubleinitialBalance)throws
	NegativeAmtException
	{
		if(initialBalance<0)
			throw new NegativeAmtException("Initial amount should not be
			negative!");
		balance = initialBalance;
		this.accountNumber = accountNumber;
	}
	public void deposit(double amount)throws NegativeAmtException
	{
		if (amount < 0)
		{
			throw new NegativeAmtException("Don't deposit negative amount!");
		}
		balance = balance + amount;
		System.out.println("Amount deposited");
		System.out.println("Balance amount : "+getBalance());
	}
	public void withdraw(double amount)throws NegativeAmtException
	{
		if (amount < 0)
		{
		throw new NegativeAmtException("Don't withdraw a negative
			amount!");
		}
		else if(amount<=balance)
		{
			balance = balance – amount;
		}
		else
		{
			System.out.println("Insufficient amount");
		}
		System.out.println("Balance amount : "+getBalance());
	}
	public double getBalance()
	{
		return balance;
	}
	public intgetAccountNumber()
	{
		return accountNumber;
	}
	public String toString ()
	{
```

```
                return "Account Number :" + accountNumber + " Balance :" + balance;
        }
}
public class UserDefinedException
{
        public static void main(String[] args)
        {
                intch,amt;
                Scanner sc=new Scanner(System.in);
                System.out.print("Enter Account Number:");
                int a=sc.nextInt();
                System.out.print("Enter the initial Amount:");
                int b=sc.nextInt();
                BankAccount ac;
                try
                {
                        ac=new BankAccount(a,b);
                        while(true)
                        {
                                System.out.println("Main Menu");
                                System.out.println("1.Deposit \n2.Withdraw \n3.Check Balance
                                \n4.Display \n5.Exit");
                                System.out.print("Enter your Choice: ");
                                ch=sc.nextInt();
                                switch(ch)
                                {
                                        case 1:
                                                System.out.print("Enter the amount to deposit:");
                                                amt=sc.nextInt();
                                                ac.deposit(amt);
                                                break;
                                        case 2:
                                                System.out.print("Enter the amount to
                                                Withdraw:");
                                                amt=sc.nextInt();
                                                ac.withdraw(amt);
                                                break;
                                        case 3:
                                                System.out.println("Balance amount :
                                                "+ac.getBalance());
                                                break;
                                        case 4:

                                                System.out.println("Your account
                                                details\n"+ac);
                                                break;
                                        case 5:
                                                sc.close();
                                                System.exit(0);
                                        default:
```

```
                                        System.out.println("Invalid Choice");
                                }
                        }
                }
                catch(NegativeAmtException e)
                {
                        System.out.println("Exception Caught : "+e);
                }
        }
}
```

OUTPUT:

```
Enter Account Number:1234
Enter the initial Amount:500
Main Menu
1.Deposit
2.Withdraw
3.Check Balance
4.Display
5.Exit
Enter your Choice: 1
Enter the amount to deposit:-456
Exception Caught : Don't deposit negative amount!
```

RESULT:

Thus the Java program to implement user defined exception handling was implementedand executed successfully.

EX.NO.: 7	WRITE A JAVA PROGRAM THAT IMPLEMENTS A MULTI-THREADED APPLICATION THAT HAS THREE THREADS. FIRST THREAD GENERATES A RANDOM INTEGER EVERY 1 SECOND AND IF THE VALUE IS EVEN, THE SECOND THREAD COMPUTES THE SQUARE OF THE NUMBER AND PRINTS. IF THE VALUE IS ODD, THE THIRD THREAD WILL PRINT THE VALUE OF THE CUBE OF THE NUMBER
DATE:	

AIM:

To write a java program to implement a multi-threaded application.

ALGORITHM:

Step 1: Start the program.
Step 2: Create a class **even** which implements first thread that computes the square of the number.
Step 3: Therun() method implements the code to be executed when thread gets executed.
Step 4: Create a class **odd** which implements second thread that computes the cube of the number.
Step 5: Create a third thread that generates random number. If the random number is even, it displaysthe square of the number. If the random number generated is odd, it displays the cube of thegiven number.
Step 6: The Multithreading is performed and the task switched between multiple threads.
Step 7: The sleep () method makes the thread to suspend for the specified time.
Step 8: Stop the program.

PROGRAM:

MultiThread.java

```
import java.util.*;
class even implements Runnable
{
        public int x;
        public even(int x)
        {
                this.x = x;
        }
        public void run()
        {
                System.out.println("New Thread "+ x +" is EVEN and Square of " + x + " is: "
                + x * x);
        }
}
```

```
class odd implements Runnable
{
	public int x;
	public odd(int x)
	{
		this.x = x;
	}
	public void run()
	{
		System.out.println("New Thread "+ x +" is ODD and Cube of " + x + " is: " +
		x * x * x);
	}
}
class A extends Thread
{
	public void run()
	{
		intnum = 0;
		Random r = new Random();
		try
		{
			for (inti = 0; i< 5; i++)
			{
				num = r.nextInt(100);
				System.out.println("Main Thread and Generated Number is " +
				num);
				if (num % 2 == 0)
				{
					Thread t1 = new Thread(new even(num));
					t1.start();
				}
				else
				{
					Thread t2 = new Thread(new odd(num));
					t2.start();
				}
				Thread.sleep(1000);
				System.out.println("---------------------------------------");
			}
		}
		catch (Exception ex)
		{
			System.out.println(ex.getMessage());
		}
```

```
        }
}
public classMultiThread
{
        public static void main(String[] args)
        {
                A a = new A();
                a.start();
        }
}
```

OUTPUT:

```
Main Thread and Generated Number is 33
New Thread 33 is ODD and Cube of 33 is: 35937
---------------------------------------
Main Thread and Generated Number is 31
New Thread 31 is ODD and Cube of 31 is: 29791
---------------------------------------
Main Thread and Generated Number is 25
New Thread 25 is ODD and Cube of 25 is: 15625
---------------------------------------
Main Thread and Generated Number is 43
New Thread 43 is ODD and Cube of 43 is: 79507
---------------------------------------
Main Thread and Generated Number is 14
New Thread 14 is EVEN and Square of 14 is: 196
---------------------------------------
```

RESULT:

Thus, the java program to implement multithreaded application was executed successfully.

EX.NO.: 8	WRITE A PROGRAM TO PERFORM FILE OPERATIONS
DATE:	

AIM:

To write a java program to copy the contents of one file to another file using file operations.

ALGORITHM:

Step 1: Start the program.

Step 2: Create a class FileCopy. Get the source and destination file names from the user.

Step 3: Create object of FileInputStream by passing the source file name.

Step 4: Using read() method read the contents of the file till end of the file.

Step 5: Create object of FileOutputStream by passing the second file to the constructor.

Step 6: Use write() method to write the contents to the destination file.

Step 7: Close the file input and output stream using close() method.

Step 8: Display the contents of both files.

Step 9: Stop the program.

PROGRAM:

FileCopy.java

```
import java.io.*;
class CopyFile
{
        public static void main(String args[]) throws IOException
        {
                inti;
                FileInputStream fin = null;
                FileOutputStreamfout = null;
                if(args.length != 2)
                {
                        System.out.println("Usage: CopyFile from to");
                        return;
                }
                System.out.println("Displaying contents of "+ args[0]+"\n");
                try
                {
                        fin = new FileInputStream(args[0]);
                        do
```

```
        {
                i = fin.read();
                if(i != -1)
                        System.out.print((char) i);
        } while(i != -1);
}
catch(IOException e)
{
        System.out.println("Error Reading File");
}
finally
{
        try
        {
                fin.close();
        }
        catch(IOException e)
        {
                System.out.println("Error Closing File");
        }
}
System.out.println("\nCopying contents of "+ args[0] + "to " + args[1]+"\n");
try
{
        fin = new FileInputStream(args[0]);
        fout = new FileOutputStream(args[1]);
        do
        {
                i = fin.read();
                if(i != -1) fout.write(i);
        } while(i != -1);
}
catch(IOException e)
{
        System.out.println("I/O Error: " + e);
}
finally
{
        try
        {
                if(fin != null)
                        fin.close();
        }
        catch(IOException e2)
```

```
                {
                        System.out.println("Error Closing Input File");
                }
                try
                {
                        if(fout != null)
                                fout.close();
                }
                catch(IOException e2)
                {
                        System.out.println("Error Closing Output File");
                }
        }
        System.out.println("\nFile Copied\n");
        System.out.println("\nDisplaying contents of "+ args[1]+"\n");
        try
        {
                fin = new FileInputStream(args[1]);
                do
                {
                        i = fin.read();
                        if(i != -1)
                                System.out.print((char) i);
                } while(i != -1);
        }
        catch(IOException e)
        {
                System.out.println("Error Reading File");
        }
        finally
        {
                try
                {
                        fin.close();
                }
                catch(IOException e)
                {
                        System.out.println("Error Closing File");
                }
        }
    }
}
```

OUTPUT:

```
R:\oop>javac CopyFile.java
R:\oop>java  CopyFile FIRST.txt SECOND.txt

Displaying contents of FIRST.txt

To use this program, specify the name
of the source file and the destination file.
For example, to copy a file called FIRST.TXT
to a file called SECOND.TXT, use the following
command line.
java CopyFile FIRST.TXT SECOND.TXT

Copying contents of FIRST.txt to SECOND.txt

File Copied

Displaying contents of SECOND.txt

To use this program, specify the name
of the source file and the destination file.
For example, to copy a file called FIRST.TXT
to a file called SECOND.TXT, use the following
command line.
java CopyFile FIRST.TXT SECOND.TXT

R:\oop>
```

RESULT:

Thus, the java program to copy the contents of one file to another file using file was written, executed and verified.

EX.NO.: 9	DEVELOP APPLICATIONS TO DEMONSTRATE THE FEATURES OF GENERICS CLASSES
DATE:	

AIM:

To write a java program to find the maximum and minimum value from the given type of elements using ageneric function.

ALGORITHM:

Step 1: Start the program.

Step 2: Create a class **Myclass**to implement generic class and generic methods.

Step 3: Get the set of the values belonging to specific data type.

Step 4: Create the objects of the class to hold integer, character and double values.

Step 5: Create the method to compare the values and find the maximum value stored in the array.

Step 6: Invoke the method with integer, character, double and string values. The output will be displayedbased on the data type passed to the method.

Step 7: Stop the program.

PROGRAM:

GenericsDemo.java

```
class MyClass<T extends Comparable<T>>
{
        T[] vals;
        MyClass(T[] obj)
        {
                vals = obj;
        }
        public T min()
        {
                T v = vals[0];
                for(inti=1; i<vals.length; i++)
                        if(vals[i].compareTo(v) < 0)
                                v = vals[i];
                return v;
        }
        public T max()
        {
                T v = vals[0];
                for(inti=1; i<vals.length;i++)
                        if(vals[i].compareTo(v) > 0)
                                v = vals[i];
                return v;
```

```
        }
}
class GenericsDemo
{
        public static void main(String args[])
        {
                Integer num[]={10,2,5,4,6,1};
                Character ch[]={'v','p','s','a','n','h'};
                Double d[]={20.2,45.4,71.6,88.3,54.6,10.4};
                String str[]= {"hai","how","are","you"};
                MyClass<Integer>iob = new MyClass<Integer>(num);
                MyClass<Character> cob = new MyClass<Character>(ch);
                MyClass<Double>dob = new MyClass<Double>(d);
                MyClass<String>sob=new MyClass<String>(str);
                System.out.println("Max value in num: " + iob.max());
                System.out.println("Min value in num: " + iob.min());
                System.out.println("Max value in ch: " + cob.max());
                System.out.println("Min value in ch: " + cob.min());
                System.out.println("Max value in d: " + dob.max());
                System.out.println("Min value in d: " + dob.min());
                System.out.println("Max value in str: " + sob.max());
                System.out.println("Min value in str: " + sob.min());
        }
}
```

OUTPUT:

```
Max value in num: 10
Min value in num: 1
Max value in ch: v
Min value in ch: a
Max value in d: 88.3
Min value in d: 10.4
Max value in str: you
Min value in str: are
```

RESULT:

Thus, the Java program to find the maximum and minimum value from the given type of elements was implemented using generics and executed successfully.

EX.NO.: 10(a)	DEVELOP APPLICATIONS USING JAVAFX CONTROLS
DATE:	

AIM:

To write a program to display multiple choice test question using JavaFX.

ALGORITHM:

Step 1: Start the program

Step 2: Import the necessary packages

Step 3: Create a public class that extends the Application class.

Step 4: Override the start() method, which is found in the Application class.

Step 5: Create the button and name "Submit" on the button.

Step 6: Create radio buttons and link them all to the group question1.

Step 7: Disable the submit button by default initially.

Step 8: Add event handlers to all the radio buttons

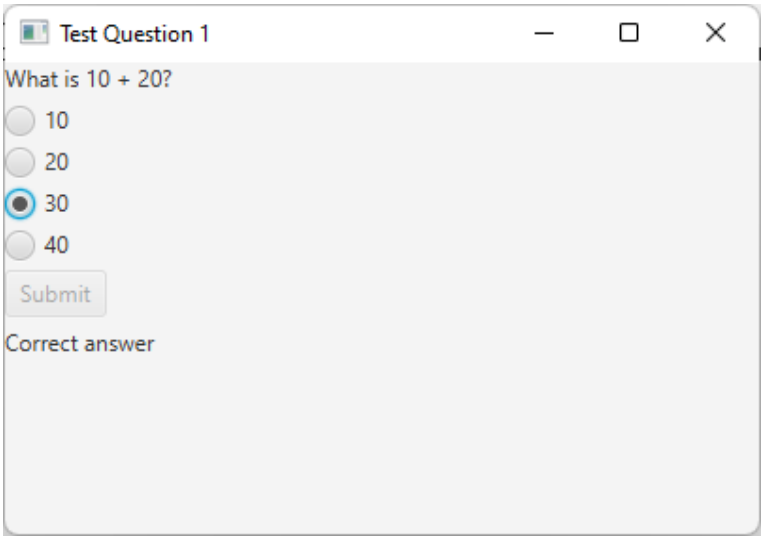
Step 9: Stop the program.

PROGRAM:

```
import javafx.application.Application;
import static javafx.application.Application.launch;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.RadioButton;
import javafx.scene.control.ToggleGroup;
import javafx.scene.layout.VBox;
import javafx.stage.Stage;
public class MCTest extends Application
{
        @Override
        public void start(Stage primaryStage)
        {
primaryStage.setTitle("Test Question 1");
                Label labelfirst= new Label("What is 10 + 20?");
                Label labelresponse= new Label();
      Button button= new Button("Submit");
                RadioButton radio1, radio2, radio3, radio4;
                radio1=new RadioButton("10");
```

```
                radio2= new RadioButton("20");
                radio3= new RadioButton("30");
                radio4= new RadioButton("40");

                ToggleGroup question1= new ToggleGroup();
                radio1.setToggleGroup(question1);
                radio2.setToggleGroup(question1);
                radio3.setToggleGroup(question1);
                radio4.setToggleGroup(question1);

                button.setDisable(true);
                radio1.setOnAction(e ->button.setDisable(false) );
                radio2.setOnAction(e ->button.setDisable(false) );
                radio3.setOnAction(e ->button.setDisable(false) );
                radio4.setOnAction(e ->button.setDisable(false) );

                button.setOnAction(e ->
                {
                        if (radio3.isSelected())
                        {
                                labelresponse.setText("Correct answer");
                                        button.setDisable(true);
                        }
                        else
                        {
                                labelresponse.setText("Wrong answer");
                                button.setDisable(true);
                        }
                });
                VBox layout= new VBox(5);
                layout.getChildren().addAll(labelfirst, radio1, radio2, radio3, radio4, button,
labelresponse);
        Scene scene1= new Scene(layout, 400, 250);
                primaryStage.setScene(scene1);
primaryStage.show();
        }
        public static void main(String[] args)
        {
                launch(args);
        }
}
```

OUTPUT:



RESULT:

Thus, the Java program for multiple choice test questions was implemented and executed successfully.

EX.NO.: 10(b)	DEVELOP APPLICATIONS USING LAYOUTS AND MENUS
DATE:	

AIM:

To write a program to create simple editor with menu using JavaFX.

ALGORITHM:

Step 1: Start the program

Step 2: Import the necessary packages

Step 3: Create a public class that extends the Application class.

Step 4: Override the start() method, which is found in the Application class.

Step 5: Create the menubar and add menu like File, Edit etc..

Step 6: In File menu add the menu items open, save and exit.

Step 7: In Edit menu add menu items like cut,copy and paste.

Step 8: Add event handlers to all menu items

Step 9: Create scene and add it to primary stage.

Step 10: Launch the application.

Step 11: Stop the program.

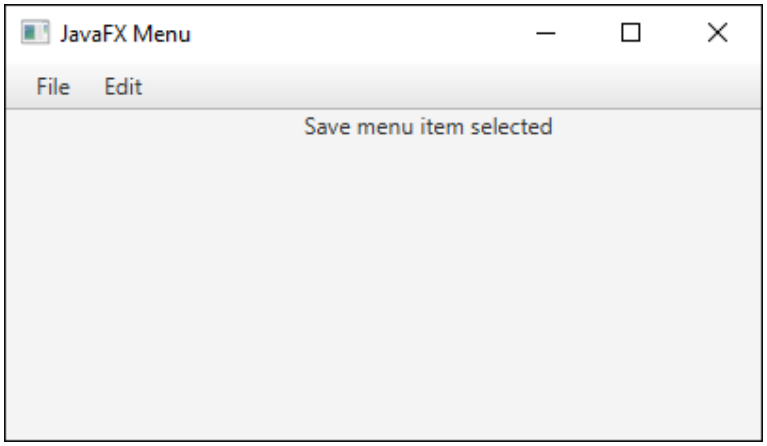
PROGRAM:

```
import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.control.Label;
import javafx.scene.control.Menu;
import javafx.scene.control.MenuBar;
import javafx.scene.control.MenuItem;
import javafx.scene.layout.VBox;
public class MenuUI extends Application {
   @Override
   public void start(Stage primaryStage) throws Exception
   {
```

```
        Menu newmenu = new Menu("File");
         Menu newmenu2 = new Menu("Edit");
        MenuItem m1 = new MenuItem("Open");
        MenuItem m2 = new MenuItem("Save");
        MenuItem m3 = new MenuItem("Exit");
        MenuItem m4 = new MenuItem("Cut");
        MenuItem m5 = new MenuItem("Copy");
        MenuItem m6 = new MenuItem("Paste");
         newmenu.getItems().add(m1);
        newmenu.getItems().add(m2);
        newmenu.getItems().add(m3);
        newmenu2.getItems().add(m4);
         newmenu2.getItems().add(m5);
        newmenu2.getItems().add(m6);
        MenuBar newmb = new MenuBar();
        newmb.getMenus().add(newmenu);
        newmb.getMenus().add(newmenu2);
        Label l = new Label("\t\t\t\t\t\t + "You have selected no menu items");
        EventHandler<ActionEvent> event = new EventHandler<ActionEvent>() {
        public void handle(ActionEvent e)
        {
                l.setText("\t\t\t\t\t\t" + ((MenuItem)e.getSource()).getText() +
                                " menu item selected");
        }
};
m1.setOnAction(event);
m2.setOnAction(event);
m3.setOnAction(event);
m4.setOnAction(event);
m5.setOnAction(event);
m6.setOnAction(event);
VBox box = new VBox(newmb,l);
Scene scene = new Scene(box,400,200);
primaryStage.setScene(scene);
```

```
primaryStage.setTitle("JavaFX Menu");
primaryStage.show();
}
   public static void main(String[] args) {
      Application.launch(args);
   }
}
```

OUTPUT:



RESULT:

Thus, the Java program for simple editor was implemented and executed successfully.

EX.NO.: 11	DEVELOP A MINI PROJECT FOR ANY APPLICATION USING JAVA CONCEPTS (LIBRARY MANAGEMENT SYSTEM)
DATE:	

Aim:

To create a library management system using java program.

Procedure:

The Library Management System has become an essential tool for public libraries, school libraries, college libraries. The Library management system helps librarians to manage, maintain and monitor the library.

There will be two module:

1. Librarian
2. User/Student

Functions of Librarian:

- Librarians can add users.
- Librarians can add books.
- Librarians can issue books for users.
- Librarians can make entries of the return books of the users.
- Librarians can view users.
- Librarians can view books.
- Librarians can view issued books.
- Librarians can view returned books.

Functions of Student/User:

- Users can check or view available books of the library
- Users can check or view his/her issued books.
- Users can check or view his/her returned books status.

Project Prerequisites:

- IDE Used: NetBeans 11.2 (you can use Eclipse)
- Java and MySql should be installed on the machine.
- Database Used: MySQL 5.5.
- To build a library management system using java we require basic knowledge of java and MySQL database operations (and JDBC).
- Java provides by default packages such as Abstract Window Toolkit (AWT) & Swing packages to create user interface (UI) for desktop applications. Also, we need two more libraries such as Mysql-JDBC-connector and java-date-picker.

Steps to Create Library Management System in Java

These are the steps to build Library Management System in Java:

1. Creating Database
2. Importing packages
3. Functions used
4. Connection to database
5. Defining Login function
6. Defining Librarian functions
7. Defining Student function

1. Create Database

In this step, we basically create our library management system database.

Also, we create four tables:

- Users table for storing information such as username, password, user_type and users table is used for login purposes.
- Books table for storing books details of library.
- Issued books table for storing the entries of the user if he/she took a book from the library.
- Return books table for storing the entries of the user if he/she returns the books to the library.

Sql Queries

a) Creating database:

```
create database library;
```

b) Creating table BOOKS

```
CREATE TABLE `books` (
`bid` int(11) NOT NULL AUTO_INCREMENT,
`book_isbn` varchar(40) NOT NULL,
`book_name` varchar(50) NOT NULL,
`book_publisher` varchar(50) NOT NULL,
`book_edition` varchar(50) NOT NULL,
`book_genre` varchar(20) NOT NULL,
`book_price` int(11) NOT NULL,
`book_pages` int(11) NOT NULL,
PRIMARY KEY (`bid`)
)
```

c) Create table ISSUED_BOOKS:

```
CREATE TABLE `issued_books` (
`IID` int(11) NOT NULL AUTO_INCREMENT,
`UID` int(11) NOT NULL,
`BID` int(11) NOT NULL,
`ISSUED_DATE` varchar(20) NOT NULL,
`PERIOD` int(11) NOT NULL,
```

```
PRIMARY KEY (`IID`),
KEY `UID` (`UID`),
KEY `BID` (`BID`),
CONSTRAINT `issued_books_ibfk_2` FOREIGN KEY (`BID`) REFERENCES `books` (`bid`),
CONSTRAINT `issued_books_ibfk_1` FOREIGN KEY (`UID`) REFERENCES `users` (`UID`))
```

d) Create table RETURNED_BOOKS:

```
CREATE TABLE `returned_books` (
`rid` int(11) NOT NULL AUTO_INCREMENT,
`bid` int(11) NOT NULL,
`uid` int(11) NOT NULL,
`return_date` varchar(50) NOT NULL,
`fine` int(11) NOT NULL DEFAULT '0',
PRIMARY KEY (`rid`),
KEY `uid` (`uid`),
KEY `bid` (`bid`),
CONSTRAINT `returned_books_ibfk_2` FOREIGN KEY (`bid`) REFERENCES `books` (`bid`),
CONSTRAINT `returned_books_ibfk_1` FOREIGN KEY (`uid`) REFERENCES `users` (`UID`))
```

e) Create table USERS:

```
CREATE TABLE `users` (
`UID` int(11) NOT NULL AUTO_INCREMENT,
`USERNAME` varchar(30) NOT NULL,
`PASSWORD` varchar(30) NOT NULL,
`USER_TYPE` int(11) NOT NULL,
PRIMARY KEY (`UID`)
)
```

f)Inserting user:

```
INSERT INTO USERS(USERNAME,PASSWORD,USER_TYPE) VALUES (“admin”,”admin”,1)
```

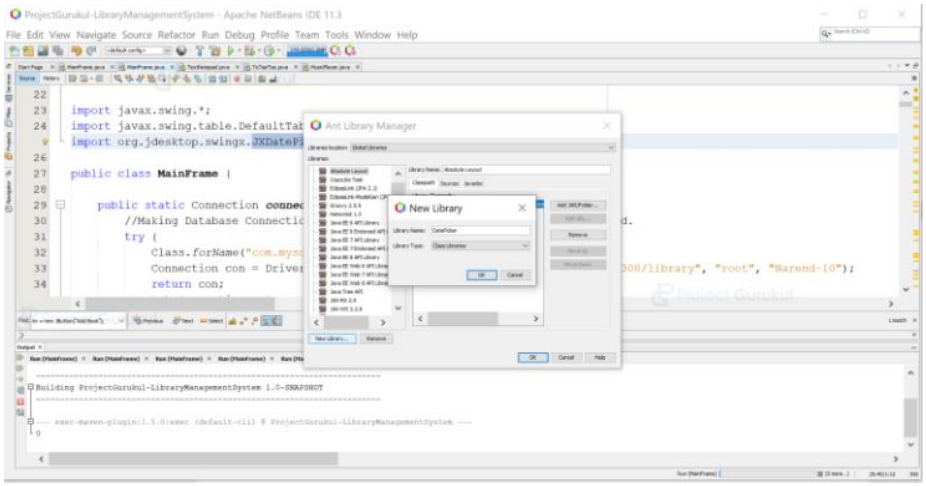
2. Importing packages

In this step, we will import required packages such as swing, awt, jxdatepicker library, mysql-connector library. etc.By default, Swing & AWT packages are installed by JAVA.

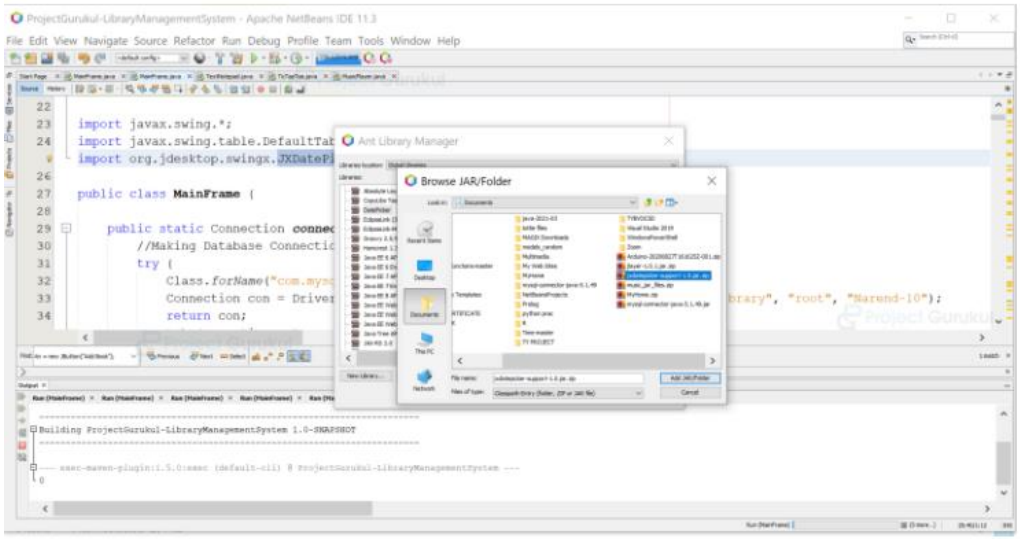
But, jxdatepicker library and MySQL-connector library we have to download and import into our library management system project.

How to import them?

Click on Tools >> Libraries >> At bottom left Click on “New Library”. Now, name your library and click on “OK”:



Click on Add JAR/Folder and select the downloaded file to add in our project. Now, add that library to java library management project.



Code:

```
import java.awt.Color;

import java.awt.GridLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;
```

```
import java.sql.*;

import java.text.DateFormat;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Date;

import java.util.concurrent.TimeUnit;

import javax.swing.*;

import javax.swing.table.DefaultTableModel;

import org.jdesktop.swingx.JXDatePicker;
```

3. Functions used

- **setLayout(layout):** This function will define the layout of the frame, window, pane, in which all the components are arranged.
- **setText("your text"):** This function will set the title or the text on the label, button, etc.
- **setVisible(true):** This function will make the frame or window visible to the user. By default, it is false.
- **setSize(int width, int height)**: This function takes two parameters such as height and width. It is used to define the size of frame/window, panel, etc.
- **setBackground(new Color(255,255,255)):** This function will set the background color of the UI component.
- **setForeground(new Color(255,255,255)):** This function will set the foreground color of the UI component.
- **add(obj):** This function is used to add the components in a sequential manner to the frame or panel.
- **executeQuery(string** query): This function will execute the sql query.
- **next():** This function will move the cursor to a new row of the table.
- **showMessageDialog():** This function will enable the dialog box to show the message on the top of the frame.
- **getText():** This function will get the text from the textfield input of the user.
- **parseInt("text"):** This function will convert the string into the integer data type.
- **setResizable(false):** This function sets the frame/window resizable. By default, it is true.
- **setColumnIdentifiers(String** names): This function will set the column names of the model of the table.
- **setOpaque(true):** This function will set up opaque so that the label component paints every pixel within its bounds.
- **addRow(Object):** This function will add the data to the new row of the model of the table.
- **isSelected():** This function will return true if the radio button is selected otherwise it will return false.
- **dispose():** This function will close the frame / window of library management system.
- **getString(int column):** This function will get the varchar/string data from the table of the mysql database.

- **getInt(int column):** This function will get the integer data from the table of the mysql database.

4. Connection to database

In this step, we will create a connection method as a reusable component, this method will connect the library management to mysql database. Make sure that you enter the proper url string of database connection such as port number, username, password, etc.

Function definitions:

forName(driverClassName): This function is used to register with the driver of the database.
getConnection(url): This function is used to make connections with databases. It takes parameters such as hostname, port number, database, username, password, etc. Generalized form: "jdbc:mysql://hostname:port/dbName", "username", "password". If your mysql does not have a password then enter "" as an empty string.

Code:

```
public static Connection connect() {

//Making Database Connection once & using multiple times whenever required.

try {

Class.forName("com.mysql.jdbc.Driver");

Connection con =

DriverManager.getConnection("jdbc:mysql://localhost:3306/

library", "root", "yourpassword");

return con;

} catch (Exception ex) {

ex.printStackTrace();

}

return null;

}
```

5. Defining Login function

In this step, we will define a login method, which will authenticate the user to java library management system. If the user type is admin then it is redirected to the librarian functions dashboard else user functions dashboard is redirected. This method will take 2 parameters such as username and password. If the user enters an empty string then it will give a message to the user to enter the correct details.

Login Code:

```
public static void loginFn() {
```

```
//Creating Login Frame
JFrame loginFrame = new JFrame("Login");
//Creating label Username
JLabel l1 = new JLabel("Username", SwingConstants.CENTER);
//Creating label Password
JLabel l2 = new JLabel("Password", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l1.setOpaque(true);
//Setting up the background color of the label.
l1.setBackground(new Color(51, 35, 85));
//Setting up the foreground color of the label.
l1.setForeground(Color.white);
//Setting up opaque so that label component paints every pixel within its bounds.
l2.setOpaque(true);
//Setting up the background color of the label.
l2.setBackground(new Color(51, 35, 85));
//Setting up the foreground color of the label.
l2.setForeground(Color.white);
//Create textfield Username
JTextField usernameTF = new JTextField();
//Setting up the background color of the textfield.
usernameTF.setBackground(new Color(51, 35, 85));
//Setting up the foreground color of the textfield.
usernameTF.setForeground(Color.white);
//Create textfield Password
JPasswordField passwordTF = new JPasswordField();
//Setting up the background color of the textfield.
passwordTF.setBackground(new Color(51, 35, 85));
//Setting up the foreground color of the textfield.
```

```
passwordTF.setForeground(Color.white);
//Create button Login
JButton loginBtn = new JButton("Login");
//Setting up the background color of the button.
loginBtn.setBackground(new Color(124, 85, 227));
//Setting up the foreground color of the button.
loginBtn.setForeground(Color.white);
//Create button cancel
JButton cancelBtn = new JButton("Cancel");
//Setting up the background color of the button.
cancelBtn.setBackground(new Color(124, 85, 227));
//Setting up the foreground color of the button.
cancelBtn.setForeground(Color.white);
//Performing action on button.
loginBtn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
String username = usernameTF.getText();
String password = passwordTF.getText();
//If username is empty
if (username.isEmpty()) {
JOptionPane.showMessageDialog(null, "Please enter username"); //Display dialog box with the message
} //If password is empty
else if (password.isEmpty()) {
JOptionPane.showMessageDialog(null, "Please enter password"); //Display dialog box with the message
} //If both the fields are present then to login the user, check whether the user exists already
else {
```

```
//Connect to the database
Connection connection = connect();
try {
Statement stmt = connection.createStatement();
String st = ("SELECT * FROM USERS WHERE USERNAME='" + username + "' AND
PASSWORD='" + password + "'"); //Retrieve username and passwords from users
ResultSet rs = stmt.executeQuery(st); //Execute query
if (rs.next() == false) { //Move pointer below
JOptionPane.showMessageDialog(null, "Invalid Username/Password!"); //Display Message
} else {
loginFrame.dispose();
rs.beforeFirst(); //Move the pointer above
while (rs.next()) {
String admin = rs.getString("user_type"); //user is admin
System.out.println(admin);
String UID = rs.getString("UID"); //Get user ID of the user
if (admin.equals("1")) { //If boolean value 1
//Redirecting to Librarian Frame
librarian_frame();
} else {
//Redirecting to User Frame for that user ID
user_frame(UID);
}
}
}
} catch (Exception ex) {
ex.printStackTrace();
}
}
```

```
}
});
cancelBtn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
loginFrame.dispose();
}
});
//Adding all login components in the login frame of java library management system.
loginFrame.add(l1);
loginFrame.add(usernameTF);
loginFrame.add(l2);
loginFrame.add(passwordTF);
loginFrame.add(loginBtn);
loginFrame.add(cancelBtn);
//Setting size of frame (width, height)
loginFrame.setSize(330, 180);//400 width and 500 height
//Setting layout of the frame
loginFrame.setLayout(new GridLayout(3, 2));
//Setting frame visible to the user
loginFrame.setVisible(true);
//Setting frame non-resizable
loginFrame.setResizable(false);
}
```

6. Defining Librarian functions

In this step, we will create the dashboard of the librarian, in which we will have 8 buttons such as add user, add book, issue book, return book, view users, view books, view issued books, view returned books. Each button will have its own action listeners to perform its own task.

Dashboard Functions:

1. Add user: The librarian will enter the details such as username, password, user type of the user and user will be added.
2. Add books: The librarian will enter the details of the new book such as isbn, book name, book publisher, price, pages, edition, etc. The details will be entered into the database and the book will be added.
3. Issue book: The librarian will enter the details of the issue book of the user such as book id, user id, date at which book was issued, etc. The details will be entered into the database and the book will be issued.
4. Return book: The librarian will enter the details of the return book of the user such as book id, user id, date at which book was returned, fine(if any), etc. The details will be entered into the java library management system.
5. View users: The librarian can view details of the users anytime.
6. View books: The librarian can view details of the books of the library anytime.
7. View issued books: The librarian can view details of the issued books anytime.
8. View returned books: The librarian can view details of the returned books anytime.

Code:

```
public static void librarian_frame() {

//Creating Librarian Frame

JFrame librarianFrame = new JFrame("Librarian Functions");

//Creating Button

JButton view_books_btn = new JButton("View Books");

//Setting up the background color of the button.

view_books_btn.setBackground(new Color(51, 35, 85));

//Setting up the foreground color of the button.

view_books_btn.setForeground(Color.white);

//Performing actions on button.

view_books_btn.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

//Creating frame.

JFrame viewBooksFrame = new JFrame("Books Available");

//Connection to Database

Connection connection = connect();

//Query for retrieving data from database
```

```
String sql = "select * from BOOKS";
try {
//Creating Statement
Statement stmt = connection.createStatement();
//Executing query
ResultSet rs = stmt.executeQuery(sql);
//Creating Table for to data will be in table format
JTable book_list = new JTable();
String[] bookColumnNames = {"Book ID", "Book ISBN", "Book Name", "Book Publisher", "Book Edition", "Book Genre", "Book price", "Book Pages"};
//Creating model for the table
DefaultTableModel bookModel = new DefaultTableModel();
//Setting up the columns names of the model
bookModel.setColumnIdentifiers(bookColumnNames);
//Adding model to the table component
book_list.setModel(bookModel);
//Setting background colour of the table
book_list.setBackground(new Color(51, 35, 85));
//Setting foreground colour of the table
book_list.setForeground(Color.white);
//Setting up table auto-resizable
book_list.setAutoResizeMode(JTable.AUTO_RESIZE_ALL_COLUMNS);
book_list.setFillsViewportHeight(true);
book_list.setFocusable(false);
//Creating scrollbars for table
JScrollPane scrollBook = new JScrollPane(book_list);
scrollBook.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);
scrollBook.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED);
```

```
while (rs.next()) {
//Fetching the data from mysql database
int book_id = rs.getInt(1);
String book_isbn = rs.getString(2);
String book_name = rs.getString(3);
String book_publisher = rs.getString(4);
String book_edition = rs.getString(5);
String book_genre = rs.getString(6);
int book_price = rs.getInt(7);
int book_pages = rs.getInt(8);
//Adding fetched data in model
bookModel.addRow(new Object[]{book_id, book_isbn, book_name, book_publisher,
book_edition, book_genre, book_price, book_pages});
}
//Adding scrollbars in the frame
viewBooksFrame.add(scrollBook);
//Setting up the size of the frame (width,height)
viewBooksFrame.setSize(800, 400);
//Setting up frame visible for user
viewBooksFrame.setVisible(true);
} catch (Exception e1) {
//Creating Dialog box to show any error if occured!
JOptionPane.showMessageDialog(null, e1);
}
}
});
//Creating button
JButton view_users_btn = new JButton("View Users");
//Setting Background color of the button.
```

```
view_users_btn.setBackground(new Color(51, 35, 85));
//Setting Foreground color of the button.
view_users_btn.setForeground(Color.white);
//Performing actions on the button.
view_users_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
//Creating frame.
JFrame viewUsersFrame = new JFrame("Users List");
//Connection to database
Connection connection = connect();
//Query for retrieving data from database
String sql = "select * from users";
try {
//Creating Statement
Statement stmt = connection.createStatement();
//Executing query
ResultSet rs = stmt.executeQuery(sql);
//Creating Table for to data will be in table format
JTable users_list = new JTable();
String[] userColumnNames = {"User ID", "User Name", "User Type"};
//Creating model for the table
DefaultTableModel userModel = new DefaultTableModel();
//Setting up the columns names of the model
userModel.setColumnIdentifiers(userColumnNames);
//Adding model to the table component
users_list.setModel(userModel);
//Setting up table auto-resizable
users_list.setAutoResizeMode(JTable.AUTO_RESIZE_ALL_COLUMNS);
```

```
users_list.setFillsViewportHeight(true);
//Setting background colour of the table.
users_list.setBackground(new Color(51, 35, 85));
//Setting foreground colour of the table.
users_list.setForeground(Color.white);
//Creating scrollbars for table
JScrollPane scrollUser = new JScrollPane(users_list);
scrollUser.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);
scrollUser.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED);
while (rs.next()) {
//Fetching the data from mysql database
int uid = rs.getInt(1);
String user_name = rs.getString(2);
int user_type = rs.getInt(4);
if (user_type == 1) {
//Checking if it is 1 then it is admin
userModel.addRow(new Object[]{uid, user_name, "ADMIN"});
} else {
//Else it will be user
userModel.addRow(new Object[]{uid, user_name, "USER"});
}
}
//Adding scrollbars in the frame
viewUsersFrame.add(scrollUser);
//Setting up the size of the frame (width,height)
viewUsersFrame.setSize(800, 400);
//Setting up frame visible for user
viewUsersFrame.setVisible(true);
```

```
} catch (Exception e1) {
//Creating Dialog box to show any error if occured!
JOptionPane.showMessageDialog(null, e1);
}
}
});
//Creating button
JButton view_issued_books_btn = new JButton("View Issued Books");
//Setting background colour of the button.
view_issued_books_btn.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the button.
view_issued_books_btn.setForeground(Color.white);
//Performing actions on button.
view_issued_books_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
//Creating button
JFrame issuedBooksFrame = new JFrame("Issued Books List");
//Connection to database
Connection connection = connect();
//Query for retrieving data from database
String sql = "select * from issued_books";
try {
//Creating Statement
Statement stmt = connection.createStatement();
//Executing query
ResultSet rs = stmt.executeQuery(sql);
//Creating Table for to data will be in table format
JTable issue_book_list = new JTable();
```

```
String[] issueBookColumnNames = {"Issue ID", "User ID", "Book ID", "Issue Date", "Period"};
//Creating model for the table
DefaultTableModel issuedBookModel = new DefaultTableModel();
//Setting up the columns names of the model
issuedBookModel.setColumnIdentifiers(issueBookColumnNames);
//Adding model to the table component
issue_book_list.setModel(issuedBookModel);
//Setting up table auto-resizable
issue_book_list.setAutoResizeMode(JTable.AUTO_RESIZE_ALL_COLUMNS);
issue_book_list.setFillsViewportHeight(true);
issue_book_list.setFocusable(false);
//Setting background colour of the table
issue_book_list.setBackground(new Color(51, 35, 85));
//Setting foreground colour of the table
issue_book_list.setForeground(Color.white);
//Creating scrollbars for table
JScrollPane scrollIssuedBook = new JScrollPane(issue_book_list);
scrollIssuedBook.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL_SCROLLBAR_AS_
NEEDED);
scrollIssuedBook.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_AS_NEED
ED);
while (rs.next()) {
//Fetching the data from mysql database
int iid = rs.getInt(1);
int uid = rs.getInt(2);
int bid = rs.getInt(3);
String issue_date = rs.getString(4);
int period = rs.getInt(5);
//Adding fetched data in model
```

```
issuedBookModel.addRow(new Object[]{iid, uid, bid, issue_date, period});
}
//Adding scrollbars in the frame
issuedBooksFrame.add(scrollIssuedBook);
//Setting up the size of the frame (width,height)
issuedBooksFrame.setSize(800, 400);
//Setting up frame visible for user
issuedBooksFrame.setVisible(true);
} catch (Exception e1) {
//Creating Dialog box to show any error if occured!
JOptionPane.showMessageDialog(null, e1);
}
}
});
//Creating button
JButton view_returned_books_btn = new JButton("View Returned Books");
//Setting Background Colour of the button.
view_returned_books_btn.setBackground(new Color(51, 35, 85));
//Setting Foreground Colour of the button.
view_returned_books_btn.setForeground(Color.white);
//Performing actions on the button.
view_returned_books_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
//Creating button.
JFrame returnedBooksFrame = new JFrame("Returned Books List");
//Connection between database and java library management system.
Connection connection = connect();
//Query for retrieving data from database.
```

```
String sql = "select * from returned_books";
try {
//Creating Statement.
Statement stmt = connection.createStatement();
//Executing query.
ResultSet rs = stmt.executeQuery(sql);
//Creating Table for data will be in table format.
JTable returned_book_list = new JTable();
String[] returnBookColumnNames = {"Return ID", "Book ID", "User ID", "Return Date", "Fine"};
//Creating a model for the table.
DefaultTableModel returnBookModel = new DefaultTableModel();
//Setting up the column names of the model.
returnBookModel.setColumnIdentifiers(returnBookColumnNames);
//Adding model to the table component.
returned_book_list.setModel(returnBookModel);
//Setting up the table auto-resizable.
returned_book_list.setAutoResizeMode(JTable.AUTO_RESIZE_ALL_COLUMNS);
returned_book_list.setFillsViewportHeight(true);
returned_book_list.setFocusable(false);
//Setting background colour of the table.
returned_book_list.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the table.
returned_book_list.setForeground(Color.white);
//Creating scrollbars for tables.
JScrollPane scrollReturnedBook = new JScrollPane(returned_book_list);
scrollReturnedBook.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL_SCROLLBAR_AS
_NEEDED);
scrollReturnedBook.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_AS_NEE
DED);
```

```
while (rs.next()) {
//Fetching the data from the mysql database.
int rid = rs.getInt(1);
int bid = rs.getInt(2);
int uid = rs.getInt(3);
String returned_date = rs.getString(4);
int fine = rs.getInt(5);
//Adding fetched data in model.
returnBookModel.addRow(new Object[]{rid, bid, uid, returned_date, fine});
}
//Adding scrollbars in the frame.
returnedBooksFrame.add(scrollReturnedBook);
//Setting up the size of the frame (width,height)
returnedBooksFrame.setSize(800, 400);
//Setting up frames visible for the user.
returnedBooksFrame.setVisible(true);
} catch (Exception e1) {
//Creating Dialog box to show any error if occured!
JOptionPane.showMessageDialog(null, e1);
}
}
});
//Creating button
JButton add_user_btn = new JButton("Add User");
//Setting Background Colour of the button.
add_user_btn.setBackground(new Color(51, 35, 85));
//Setting Foreground Colour of the button.
add_user_btn.setForeground(Color.white);
//Performing actions on buttons.
```

```
add_user_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
//Creating frame
JFrame add_user_frame = new JFrame("Enter User Details"); //Frame to enter user details
//Creating label
JLabel l1 = new JLabel("Username", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l1.setOpaque(true);
//Setting Background Colour of the label.
l1.setBackground(new Color(51, 35, 85));
//Setting Foreground Colour of the label.
l1.setForeground(Color.white);
//Creating label
JLabel l2 = new JLabel("Password", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l2.setOpaque(true);
//Setting Background Colour of the label.
l2.setBackground(new Color(51, 35, 85));
//Setting Foreground Colour of the label.
l2.setForeground(Color.white);
//Creating textfield
JTextField add_username_tf = new JTextField();
//Setting Background Colour of the textfield.
add_username_tf.setBackground(new Color(51, 35, 85));
//Setting Foreground Colour of the textfield.
add_username_tf.setForeground(Color.white);
//Creating textfield
JPasswordField add_password_tf = new JPasswordField();
```

```
//Setting Background Colour of the textfield.
add_password_tf.setBackground(new Color(51, 35, 85));
//Setting Foreground Colour of the textfield.
add_password_tf.setForeground(Color.white);
//Creating radio button
JRadioButton user_type_radio1 = new JRadioButton("Admin");
//Aligning center
user_type_radio1.setHorizontalAlignment(SwingConstants.CENTER);
//Setting Background Colour of the radiobutton.
user_type_radio1.setBackground(new Color(51, 35, 85));
//Setting Foreground Colour of the radiobutton.
user_type_radio1.setForeground(Color.white);
//Creating radio button
JRadioButton user_type_radio2 = new JRadioButton("User");
//Aligning center
user_type_radio2.setHorizontalAlignment(SwingConstants.CENTER);
//Setting Background Colour of the radiobutton.
user_type_radio2.setBackground(new Color(51, 35, 85));
//Setting Foreground Colour of the radiobutton.
user_type_radio2.setForeground(Color.white);
//Adding radio buttons in buttongroup
ButtonGroup user_type_btn_grp = new ButtonGroup();
user_type_btn_grp.add(user_type_radio1);
user_type_btn_grp.add(user_type_radio2);
//Creating button.
JButton create_btn = new JButton("Create");
//Setting Background Colour of the button.
create_btn.setBackground(new Color(124, 85, 227));
//Setting Foreground Colour of the button.
```

```
create_btn.setForeground(Color.white);
//Creating button.
JButton user_entry_cancel_btn = new JButton("Cancel");
//Setting Background Colour of the button.
user_entry_cancel_btn.setBackground(new Color(124, 85, 227));
//Setting Foreground Colour of the button.
user_entry_cancel_btn.setForeground(Color.white);
//Performing actions on the button.
create_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
//Getting data from the textfield.
String username = add_username_tf.getText();
String password = add_password_tf.getText();
//Connection to database.
Connection connection = connect();
try {
//Creating statement
Statement stmt = connection.createStatement();
//Check if radio1 is click or not
//If radio1 is click then it is added as admin, else normal student
if (user_type_radio1.isSelected()) {
//Query to insert inside in the table
stmt.executeUpdate("INSERT INTO USERS(USERNAME,PASSWORD,USER_TYPE) VALUES
('" + username + "','" + password + "','" + "1" + "')");
//Creating Dialog Box to display message.
JOptionPane.showMessageDialog(null, "Admin added!");
add_user_frame.dispose();
} else {
```

```
//Query to insert inside in the table.
stmt.executeUpdate("INSERT INTO USERS(USERNAME,PASSWORD,USER_TYPE) VALUES
('" + username + "','" + password + "','" + "0" + "')");
//Creating Dialog Box to display message.
JOptionPane.showMessageDialog(null, "User added!");
add_user_frame.dispose();
}
} catch (Exception e1) {
//Creating Dialog box to show any error if occured!
JOptionPane.showMessageDialog(null, e1);
}
}
});
//Performing actions on button.
user_entry_cancel_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
add_user_frame.dispose();
}
});
//Adding components in frame.
add_user_frame.add(l1);
add_user_frame.add(add_username_tf);
add_user_frame.add(l2);
add_user_frame.add(add_password_tf);
add_user_frame.add(user_type_radio1);
add_user_frame.add(user_type_radio2);
add_user_frame.add(create_btn);
add_user_frame.add(user_entry_cancel_btn);
```

```
//Setting up the size of the frame (width,height)
add_user_frame.setSize(350, 200);
//Setting up layout of the frame
add_user_frame.setLayout(new GridLayout(4, 2));
//Setting up the frame visible
add_user_frame.setVisible(true);
//Setting up the table auto-resizable.
add_user_frame.setResizable(false);
}
});
//Creating button.
JButton add_book_btn = new JButton("Add Book");
//Setting Background Colour of the button.
add_book_btn.setBackground(new Color(51, 35, 85));
//Setting Foreground Colour of the button.
add_book_btn.setForeground(Color.white);
//Performing actions on button.
add_book_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
//Creating Frame.
JFrame book_frame = new JFrame("Enter Book Details");
//Creating labels
JLabel l1, l2, l3, l4, l5, l6, l7;
l1 = new JLabel("ISBN", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l1.setOpaque(true);
//Setting background colour of the label.
l1.setBackground(new Color(51, 35, 85));
```

```
//Setting the foreground colour of the label.
l1.setForeground(Color.white);
l2 = new JLabel("Name", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l2.setOpaque(true);
//Setting background colour of the label.
l2.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the label.
l2.setForeground(Color.white);
l3 = new JLabel("Publisher", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l3.setOpaque(true);
//Setting background colour of the label.
l3.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the label.
l3.setForeground(Color.white);
l4 = new JLabel("Edition", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l4.setOpaque(true);
//Setting background colour of the label.
l4.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the label.
l4.setForeground(Color.white);
l5 = new JLabel("Genre", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l5.setOpaque(true);
//Setting background colour of the label.
l5.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the label.
```

```
l5.setForeground(Color.white);
l6 = new JLabel("Price", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l6.setOpaque(true);
//Setting background colour of the label.
l6.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the label.
l6.setForeground(Color.white);
l7 = new JLabel("Pages", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l7.setOpaque(true);
//Setting background colour of the label.
l7.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the label.
l7.setForeground(Color.white);
//Creating textfield.
JTextField book_isbn_tf = new JTextField();
//Setting background colour of the textfield.
book_isbn_tf.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the textfield.
book_isbn_tf.setForeground(Color.white);
//Creating textfield
JTextField book_name_tf = new JTextField();
//Setting background colour of the textfield.
book_name_tf.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the textfield.
book_name_tf.setForeground(Color.white);
//Creating textfield.
JTextField book_publisher_tf = new JTextField();
```

```
//Setting background colour of the textfield.
book_publisher_tf.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the textfield.
book_publisher_tf.setForeground(Color.white);
//Creating textfield.
JTextField book_edition_tf = new JTextField();
//Setting background colour of the textfield.
book_edition_tf.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the textfield.
book_edition_tf.setForeground(Color.white);
//Creating textfield.
JTextField book_genre_tf = new JTextField();
//Setting background colour of the textfield.
book_genre_tf.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the textfield.
book_genre_tf.setForeground(Color.white);
//Creating textfield.
JTextField book_price_tf = new JTextField();
//Setting background colour of the textfield.
book_price_tf.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the textfield.
book_price_tf.setForeground(Color.white);
//Creating textfield.
JTextField book_pages_tf = new JTextField();
//Setting background colour of the textfield.
book_pages_tf.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the textfield.
book_pages_tf.setForeground(Color.white);
//Creating button.
```

```
JButton create_btn = new JButton("Submit");
//Setting background colour of the button.
create_btn.setBackground(new Color(124, 85, 227));
//Setting the foreground colour of the button.
create_btn.setForeground(Color.white);
//Creating button.
JButton add_book_cancel_btn = new JButton("Cancel");
//Setting background colour of the button.
add_book_cancel_btn.setBackground(new Color(124, 85, 227));
//Setting the foreground colour of the button.
add_book_cancel_btn.setForeground(Color.white);
//Performing actions on the button.
create_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
//Getting data from the textfield.
String book_isbn = book_isbn_tf.getText();
String book_name = book_name_tf.getText();
String book_publisher = book_publisher_tf.getText();
String book_edition = book_edition_tf.getText();
String book_genre = book_genre_tf.getText();
//Converting bookprice and bookpages to integer from string.
int book_price = Integer.parseInt(book_price_tf.getText());
int book_pages = Integer.parseInt(book_pages_tf.getText());
//Connection to database.
Connection connection = connect();
try {
//Creating statement
Statement stmt = connection.createStatement();
```

```
//Query to insert in the table.
stmt.executeUpdate("INSERT INTO
BOOKS(book_isbn,book_name,book_publisher,book_edition,book_genre,book_price,book_pages)
"
+ " VALUES ('" + book_isbn + "','" + book_name + "','" + book_publisher + "','" + book_edition +
"','" + book_genre + "','" + book_price + "'," + book_pages + ")");
//Creating Dialog Box to display message.
JOptionPane.showMessageDialog(null, "Book added!");
book_frame.dispose();
} catch (Exception e1) {
//Creating Dialog box to show any error if occured!
JOptionPane.showMessageDialog(null, e1);
}
}
});
//Performing actions on the button.
add_book_cancel_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
book_frame.dispose();
}
});
//Adding components in the frame.
book_frame.add(l1);
book_frame.add(book_isbn_tf);
book_frame.add(l2);
book_frame.add(book_name_tf);
book_frame.add(l3);
book_frame.add(book_publisher_tf);
book_frame.add(l4);
```

```
book_frame.add(book_edition_tf);
book_frame.add(l5);
book_frame.add(book_genre_tf);
book_frame.add(l6);
book_frame.add(book_price_tf);
book_frame.add(l7);
book_frame.add(book_pages_tf);
book_frame.add(create_btn);
book_frame.add(add_book_cancel_btn);
//Setting up the size of the frame (width,height)
book_frame.setSize(800, 500);
//Setting up layout of the frame
book_frame.setLayout(new GridLayout(8, 2));
//Setting up the frame visible
book_frame.setVisible(true);
//Setting up the table auto-resizable.
book_frame.setResizable(false);
}
});
//Creating button
JButton add_issue_book_btn = new JButton("Issue Book");
//Setting background colour of the button.
add_issue_book_btn.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the button.
add_issue_book_btn.setForeground(Color.white);
//Performing actions on the button.
add_issue_book_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
```

```
//Creating frame.
JFrame issue_book_frame = new JFrame("Enter Details");
//Creating panel.
JPanel pickerPanel = new JPanel();
//Creating a datepicker.
JXDatePicker picker = new JXDatePicker();
//Setting up current date in datepicker
picker.setDate(Calendar.getInstance().getTime());
//Formatting datepicker.
picker.setFormats(new SimpleDateFormat("dd.MM.yyyy"));
//Adding datepicker in the panel.
pickerPanel.add(picker);
//Setting background colour of the panel.
pickerPanel.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the panel.
pickerPanel.setForeground(Color.white);
//Creating labels
JLabel l1, l2, l3, l4;
l1 = new JLabel("Book ID", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l1.setOpaque(true);
//Setting background colour of the label.
l1.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the label.
l1.setForeground(Color.white);
l2 = new JLabel("User/Student ID", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l2.setOpaque(true);
//Setting background colour of the label.
```

```
l2.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the label.
l2.setForeground(Color.white);
l3 = new JLabel("Period(days)", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l3.setOpaque(true);
//Setting background colour of the label.
l3.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the label.
l3.setForeground(Color.white);
l4 = new JLabel("Issued Date(DD-MM-YYYY)", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l4.setOpaque(true);
//Setting background colour of the label.
l4.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the label.
l4.setForeground(Color.white);
//Creating textfield.
JTextField bid_tf = new JTextField();
//Setting background colour of the textfield.
bid_tf.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the textfield.
bid_tf.setForeground(Color.white);
//Creating textfield.
JTextField uid_tf = new JTextField();
//Setting background colour of the textfield.
uid_tf.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the textfield.
uid_tf.setForeground(Color.white);
```

```
//Creating textfield.
JTextField period_tf = new JTextField();
//Setting background colour of the textfield.
period_tf.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the textfield.
period_tf.setForeground(Color.white);
//Creating button.
JButton create_btn = new JButton("Submit");
//Setting background colour of the button.
create_btn.setBackground(new Color(124, 85, 227));
//Setting the foreground colour of the button.
create_btn.setForeground(Color.white);
//Creating button.
JButton issue_book_cancel_btn = new JButton("Cancel");
//Setting background colour of the button.
issue_book_cancel_btn.setBackground(new Color(124, 85, 227));
//Setting the foreground colour of the button.
issue_book_cancel_btn.setForeground(Color.white);
//Performing actions on the button.
create_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
//Getting data from textfield.
int uid = Integer.parseInt(uid_tf.getText());
int bid = Integer.parseInt(bid_tf.getText());
String period = period_tf.getText();
Date oDate = picker.getDate();
//Formatting date.
DateFormat oDateFormat = new SimpleDateFormat("dd-MM-yyyy");
```

```
String issued_date = oDateFormat.format(oDate);
//Converting period from string to integer.
int period_int = Integer.parseInt(period);
//Connection to the database
Connection connection = connect();
try {
//Creating Statement
Statement stmt = connection.createStatement();
//Query to insert data in the table.
stmt.executeUpdate("INSERT INTO issued_books(UID,BID,ISSUED_DATE,PERIOD) VALUES
('" + uid + "','" + bid + "','" + issued_date + "'," + period_int + ")");
//Creating Dialog Box to display message.
JOptionPane.showMessageDialog(null, "Book Issued!");
issue_book_frame.dispose();
} catch (Exception e1) {
//Creating Dialog box to show any error if occured!
JOptionPane.showMessageDialog(null, e1);
}
}
});
issue_book_cancel_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
issue_book_frame.dispose();
}
});
//Adding components in the frame
issue_book_frame.add(l1);
issue_book_frame.add(bid_tf);
```

```
issue_book_frame.add(l2);
issue_book_frame.add(uid_tf);
issue_book_frame.add(l3);
issue_book_frame.add(period_tf);
issue_book_frame.add(l4);
issue_book_frame.getContentPane().add(pickerPanel);
issue_book_frame.add(create_btn);
issue_book_frame.add(issue_book_cancel_btn);
//Setting up the size of the frame (width,height)
issue_book_frame.setSize(600, 300);
//Setting up frame layout
issue_book_frame.setLayout(new GridLayout(5, 2));
//Setting up the frame visible
issue_book_frame.setVisible(true);
//Setting up table auto-resizable.
issue_book_frame.setResizable(false);
}
});
//Creating button.
JButton add_return_book_btn = new JButton("Return Book");
//Setting background colour of the button.
add_return_book_btn.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the button.
add_return_book_btn.setForeground(Color.white);
//Performing actions on the button.
add_return_book_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
//Creating frame.
```

```
JFrame returnBookFrame = new JFrame("Enter Details");
//Creating the labels.
JLabel l1 = new JLabel("Book ID", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l1.setOpaque(true);
//Setting background colour of the label.
l1.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the label.
l1.setForeground(Color.white);
JLabel l2 = new JLabel("User ID", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l2.setOpaque(true);
//Setting background colour of the label.
l2.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the label.
l2.setForeground(Color.white);
//Creating labels.
JLabel l3 = new JLabel("Return Date(DD-MM-YYYY)", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l3.setOpaque(true);
//Setting background colour of the label.
l3.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the label.
l3.setForeground(Color.white);
JLabel l4 = new JLabel("Fine", SwingConstants.CENTER);
//Setting up opaque so that label component paints every pixel within its bounds.
l4.setOpaque(true);
//Setting background colour of the label.
l4.setBackground(new Color(51, 35, 85));
```

```
//Setting the foreground colour of the label.
l4.setForeground(Color.white);
//Creating textfield.
JTextField bid_tf = new JTextField();
//Setting background colour of the textfield.
bid_tf.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the textfield.
bid_tf.setForeground(Color.white);
//Creating textfield.
JTextField uid_tf = new JTextField();
//Setting background colour of the textfield.
uid_tf.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the textfield.
uid_tf.setForeground(Color.white);
//Creating panel for date.
JPanel pickerPanel = new JPanel();
//Creating a datepicker.
JXDatePicker picker = new JXDatePicker();
//Getting and Setting up the current time of the date.
picker.setDate(Calendar.getInstance().getTime());
//Setting up the format of the date picker.
picker.setFormats(new SimpleDateFormat("dd.MM.yyyy"));
//Creating textfield.
JTextField fine_tf = new JTextField();
//Setting background colour of the textfield.
fine_tf.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the textfield.
fine_tf.setForeground(Color.white);
//Adding datepicker in panel.
```

```
pickerPanel.add(picker);
//Setting background colour of the panel.
pickerPanel.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the panel.
pickerPanel.setForeground(Color.white);
//Creating button.
JButton return_book_btn = new JButton("Return");
//Setting background colour of the button.
return_book_btn.setBackground(new Color(124, 85, 227));
//Setting the foreground colour of the button.
return_book_btn.setForeground(Color.white);
//Creating button.
JButton cancel_book_btn = new JButton("Cancel");
//Setting background colour of the button.
cancel_book_btn.setBackground(new Color(124, 85, 227));
//Setting the foreground colour of the button.
cancel_book_btn.setForeground(Color.white);
//Performing actions on button.
return_book_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
//Getting data from text fields.
int bid = Integer.parseInt(bid_tf.getText());
int uid = Integer.parseInt(uid_tf.getText());
int fine = Integer.parseInt(fine_tf.getText());
Date oDate = picker.getDate();
//Formatting Date.
DateFormat oDateFormat = new SimpleDateFormat("dd-MM-yyyy");
String return_date = oDateFormat.format(oDate);
```

```
try {
//Connection to database
Connection connection = connect();
//Creating Statement
Statement stmt = connection.createStatement();
//Querying to insert in the table.
stmt.executeUpdate("INSERT INTO returned_books(bid,uid,return_date,fine) VALUES ('" + bid + "','" + uid + "','" + return_date + "'," + fine + ")");
//Creating Dialog Box to display message.
JOptionPane.showMessageDialog(null, "Book Returned!");
returnBookFrame.dispose();
} catch (Exception e1) {
//Creating Dialog box to show any error if occured!
JOptionPane.showMessageDialog(null, e1);
}
}
});
//Performing actions on the button.
cancel_book_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
returnBookFrame.dispose();
}
});
//Adding all return book components in the frame
returnBookFrame.add(l1);
returnBookFrame.add(bid_tf);
returnBookFrame.add(l2);
returnBookFrame.add(uid_tf);
```

```
returnBookFrame.add(l3);
returnBookFrame.getContentPane().add(pickerPanel);
returnBookFrame.add(l4);
returnBookFrame.add(fine_tf);
returnBookFrame.add(return_book_btn);
returnBookFrame.add(cancel_book_btn);
//Setting up the size of the frame
returnBookFrame.setSize(600, 300);
//Setting up the layout of the frame
returnBookFrame.setLayout(new GridLayout(5, 2));
//Setting up the frame visible
returnBookFrame.setVisible(true);
//Setting up frame non-resizable
returnBookFrame.setResizable(false);
}
});
//Setting the layout of Librarian Frame
librarianFrame.setLayout(new GridLayout(2, 4));
//Adding Librarian components in the Librarian Frame
librarianFrame.add(add_user_btn);
librarianFrame.add(add_book_btn);
librarianFrame.add(add_issue_book_btn);
librarianFrame.add(add_return_book_btn);
librarianFrame.add(view_users_btn);
librarianFrame.add(view_books_btn);
librarianFrame.add(view_issued_books_btn);
librarianFrame.add(view_returned_books_btn);
//Setting size of the frame (width,height)
librarianFrame.setSize(800, 200);
```

```
//Setting up the frame visible to the user

librarianFrame.setVisible(true);

//Setting up frame non-resizable

librarianFrame.setResizable(false);

}
```

7. Defining Student functions

In this step, we will create the dashboard of the student/user, in which we will have 3 buttons such as view books, view issued books, view returned books. Each button will have its own action listeners to perform its own task.

Dashboard Functions:

1. View books: The student can view details of the books of the library anytime.
2. View issued books: The student can check and view which books he/she has issued in the java library management system.
3. View returned books: The student can check and view which books he/she has returned to the library.

Code:

```
public static void user_frame(String UID) {

//Creating Frame for Student

JFrame studentFrame = new JFrame("Student Functions");

//Creating button

JButton view_books_btn = new JButton("View Books");

//Setting Background Colour of the button.

view_books_btn.setBackground(new Color(51, 35, 85));

//Setting Foreground Colour of the button.

view_books_btn.setForeground(Color.white);

view_books_btn.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

//Creating Frame.

JFrame viewBooksUserFrame = new JFrame("Books Available");

//Connection to database.

Connection connection = connect();
```

```
//Query for retrieving data from database.
String sql = "select * from books";
try {
//Creating Statement.
Statement stmt = connection.createStatement();
//Executing query.
ResultSet rs = stmt.executeQuery(sql);
//Creating Table for data will be in table format.
JTable book_list = new JTable();
String[] bookColumnNames = {"Book ID", "Book ISBN", "Book Name", "Book Publisher", "Book Edition", "Book Genre", "Book price", "Book Pages"};
//Creating a model for the table.
DefaultTableModel bookModel = new DefaultTableModel();
//Setting up the column names of the model.
bookModel.setColumnIdentifiers(bookColumnNames);
//Adding model to the table component.
book_list.setModel(bookModel);
//Setting background colour of the table.
book_list.setBackground(new Color(51, 35, 85));
//Setting the foreground colour of the table.
book_list.setForeground(Color.white);
//Setting up the table auto-resizable.
book_list.setAutoResizeMode(JTable.AUTO_RESIZE_ALL_COLUMNS);
book_list.setFillsViewportHeight(true);
book_list.setFocusable(false);
//Creating scrollbars for table.
JScrollPane scrollBook = new JScrollPane(book_list);
scrollBook.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);
```

```
scrollBook.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED);
while (rs.next()) {
//Fetching the data from mysql database
int book_id = rs.getInt(1);
String book_isbn = rs.getString(2);
String book_name = rs.getString(3);
String book_publisher = rs.getString(4);
String book_edition = rs.getString(5);
String book_genre = rs.getString(6);
int book_price = rs.getInt(7);
int book_pages = rs.getInt(8);
//Adding fetched data in model
bookModel.addRow(new Object[]{book_id, book_isbn, book_name, book_publisher,
book_edition, book_genre, book_price, book_pages});
}
//Adding scrollbars in the frame.
viewBooksUserFrame.add(scrollBook);
//Setting up the size of the frame. (width,height)
viewBooksUserFrame.setSize(800, 400);
//Setting up frame visible for the user.
viewBooksUserFrame.setVisible(true);
} catch (Exception e1) {
//Creating Dialog box to show any error if occured!
JOptionPane.showMessageDialog(null, e1);
}
}
}
);
//Creating Button.
```

```
JButton view_user_issued_books_btn = new JButton("Issued Books");
//Setting Background color of the button.
view_user_issued_books_btn.setBackground(new Color(51, 35, 85));
//Setting Foreground color of the button.
view_user_issued_books_btn.setForeground(Color.white);
//Performing action on the button.
view_user_issued_books_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
//Creating frame
JFrame viewUserIssuedBooksFrame = new JFrame("My Issued Books");
//Storing userid
int userid = Integer.parseInt(UID);
//Connection to database
Connection connection = connect();
//Database Query
String sql = "select issued_books.iid as iid, issued_books.bid as bid, issued_books.uid as uid,"
+ " books.book_isbn as book_isbn, books.book_name as book_name, books.book_publisher as book_publisher, "
+ "books.book_edition as book_edition, books.book_genre as book_genre, books.book_price as book_price,"
+ " books.book_pages as book_pages, issued_books.issued_date as issued_date, issued_books.period as period from books,"
+ "issued_books where books.bid=issued_books.bid and issued_books.uid=" + userid;
try {
//Creating statement
Statement stmt = connection.createStatement();
//Executing query
ResultSet rs = stmt.executeQuery(sql);
//Creating Table for to data will be in table format
```

```
JTable issued_book_list = new JTable();

String[] issuedBookColumnNames = {"Issue ID", "Book ID", "User ID", "Book ISBN", "Book Name", "Book Publisher", "Book Edition", "Book Genre", "Book Price", "Book Pages", "Issued Date", "Period"};

//Creating model for the table

DefaultTableModel bookModel = new DefaultTableModel();

//Setting up the columns names of the model

bookModel.setColumnIdentifiers(issuedBookColumnNames);

//Adding model to the table component

issued_book_list.setModel(bookModel);

//Setting background colour of the table

issued_book_list.setBackground(new Color(51, 35, 85));

//Setting foreground colour of the table

issued_book_list.setForeground(Color.white);

//Setting up table auto-resizable

issued_book_list.setAutoResizeMode(JTable.AUTO_RESIZE_ALL_COLUMNS);

issued_book_list.setFillsViewportHeight(true);

issued_book_list.setFocusable(false);

//Creating scrollbars for table

JScrollPane scrollIssuedBook = new JScrollPane(issued_book_list);

scrollIssuedBook.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);

scrollIssuedBook.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED);

while (rs.next()) {

//Fetching the data from mysql database

int iid = rs.getInt(1);

int bid = rs.getInt(2);

int uid = rs.getInt(3);

String book_isbn = rs.getString(4);
```

```
String book_name = rs.getString(5);
String book_publisher = rs.getString(6);
String book_edition = rs.getString(7);
String book_genre = rs.getString(8);
int book_price = rs.getInt(9);
int book_pages = rs.getInt(10);
String issued_date = rs.getString(11);
int period = rs.getInt(12);
//Adding fetched data in model
bookModel.addRow(new Object[]{iid, bid, uid, book_isbn, book_name, book_publisher,
book_edition, book_genre, book_price, book_pages, issued_date, period});
}
//Adding scrollbars.
viewUserIssuedBooksFrame.add(scrollIssuedBook);
//Setting up the dimensions of the frame.
viewUserIssuedBooksFrame.setSize(1200, 600);
//Setting up the frame visible.
viewUserIssuedBooksFrame.setVisible(true);
//Setting up the frame non-resizable.
viewUserIssuedBooksFrame.setResizable(false);
} catch (Exception e1) {
//Creating Dialog box to show any error if occured!
JOptionPane.showMessageDialog(null, e1);
}
}
});
//Creating Button
JButton view_user_returned_books_btn = new JButton("My Returned Books");
//Setting Background color of the button.
```

```
view_user_returned_books_btn.setBackground(new Color(51, 35, 85));
//Setting Foreground color of the button.
view_user_returned_books_btn.setForeground(Color.white);
//Performing action on the button.
view_user_returned_books_btn.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
//Creating frame
JFrame viewUserReturnedBooksFrame = new JFrame("My Returned Books");
//Storing userid
int userid = Integer.parseInt(UID);
//Connection to database
Connection connection = connect();
//Query for retrieving java library management data from database
String sql = "select returned_books.rid as rid, returned_books.bid as bid, returned_books.uid as uid,"
+ "books.book_isbn as book_isbn, books.book_name as book_name, books.book_publisher as book_publisher,"
+ "books.book_edition as book_edition, books.book_genre as book_genre, books.book_price as book_price,"
+ "books.book_pages as book_pages, returned_books.return_date as return_date, returned_books.fine as fine "
+ "from books, returned_books where books.bid=returned_books.bid and returned_books.uid=" + userid;
try {
//Creating Statement
Statement stmt = connection.createStatement();
//Executing query
ResultSet rs = stmt.executeQuery(sql);
//Creating Table for to data will be in table format
```

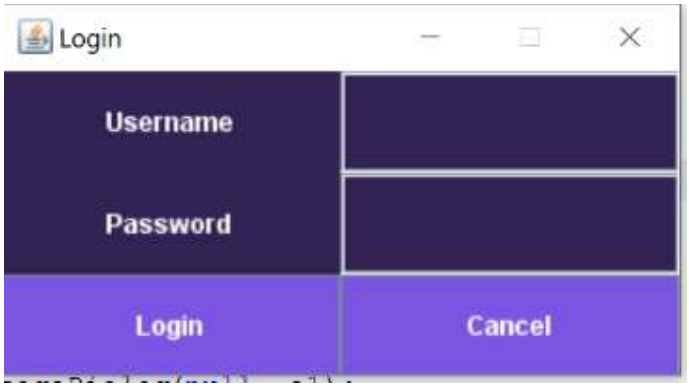
```
JTable returned_book_list = new JTable();

String[] returnedBookColumnNames = {"Return ID", "Book ID", "User ID", "Book ISBN", "Book Name", "Book Publisher", "Book Edition", "Book Genre", "Book Price", "Book Pages", "Returned Date", "Fine"};

//Creating model for the table

DefaultTableModel bookModel = new DefaultTableModel();

//Setting up the columns names of the model

bookModel.setColumnIdentifiers(returnedBookColumnNames);

//Adding model to the table component

returned_book_list.setModel(bookModel);

//Setting background colour of the table

returned_book_list.setBackground(new Color(51, 35, 85));

//Setting foreground colour of the table

returned_book_list.setForeground(Color.white);

//Setting up table auto-resizable

returned_book_list.setAutoResizeMode(JTable.AUTO_RESIZE_ALL_COLUMNS);

returned_book_list.setFillsViewportHeight(true);

returned_book_list.setFocusable(false);

//Creating scrollbars for table

JScrollPane scrollIssuedBook = new JScrollPane(returned_book_list);

scrollIssuedBook.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);

scrollIssuedBook.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED);

while (rs.next()) {

//Fetching the data from mysql database

int rid = rs.getInt(1);

int bid = rs.getInt(2);

int uid = rs.getInt(3);

String book_isbn = rs.getString(4);
```

```
String book_name = rs.getString(5);
String book_publisher = rs.getString(6);
String book_edition = rs.getString(7);
String book_genre = rs.getString(8);
int book_price = rs.getInt(9);
int book_pages = rs.getInt(10);
String returned_date = rs.getString(11);
int fine = rs.getInt(12);
//Adding fetched library management data in model
bookModel.addRow(new Object[]{rid, bid, uid, book_isbn, book_name, book_publisher,
book_edition, book_genre, book_price, book_pages, returned_date, fine});
}
//Adding scrollbars.
viewUserReturnedBooksFrame.add(scrollIssuedBook);
//Setting up the dimensions of the frame. Params:(width,height)
viewUserReturnedBooksFrame.setSize(1200, 600);
//Setting up the frame visible.
viewUserReturnedBooksFrame.setVisible(true);
//Setting up the frame is non-resizable.
viewUserReturnedBooksFrame.setResizable(false);
} catch (Exception e1) {
//Creating Dialog box to show any error if occured!
JOptionPane.showMessageDialog(null, e1);
}
}
});
//Setting Layout of the student frame.
studentFrame.setLayout(new GridLayout(3, 1));
//Adding all the components in the student frame.
```

```
studentFrame.add(view_books_btn);
studentFrame.add(view_user_issued_books_btn);
studentFrame.add(view_user_returned_books_btn);
//Setting size of the student frame (width,height)
studentFrame.setSize(500, 500);
//Setting up the frame visible
studentFrame.setVisible(true);
//Setting up frame non-resizable
studentFrame.setResizable(false);
}
```

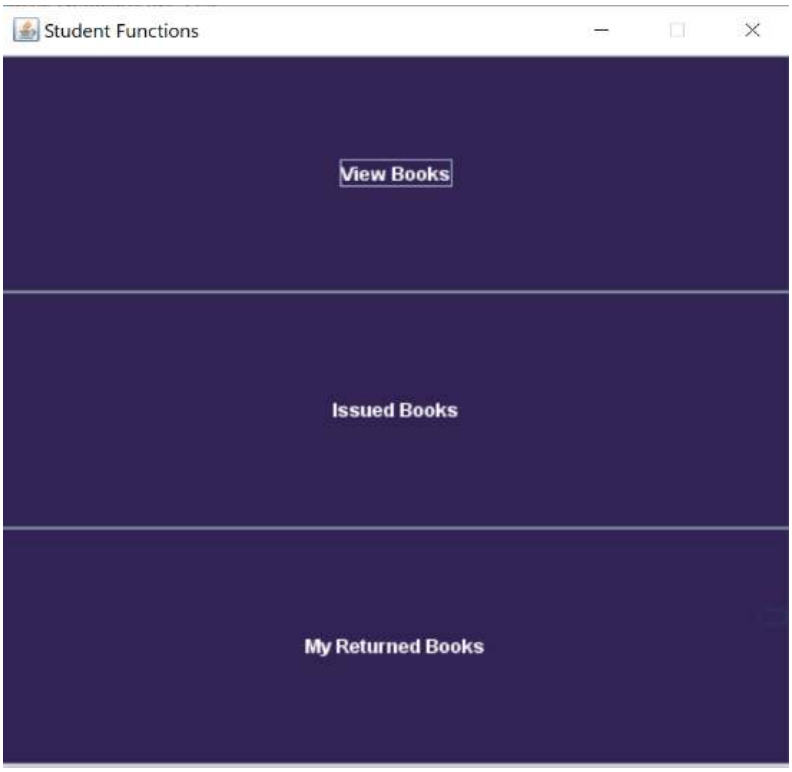
Screen Shorts:

Login Page



Librarian Functions:

User / Student functions dashboard:



Conclusion:

We have finally built our Library Management System using Java and MySQL. Now librarians can add users, books, issue books, return books, and also they can view users, books, issued books, returned books. Students/Users can view available books of the library, also users can check which book his/her issued books and returned books. From this java project, we have also learned how we can connect a MySQL database with java, and also how to query a database via Java program.