JAVA ASSIGNMENT 1

1. Design a class named Fan to represent a fan. The class contains:

Three constants named SLOW, MEDIUM, and FAST with values 1, 2, and 3 to denote the fan speed.

An int data field named speed that specifies the speed of the fan (default SLOW).

A boolean data field named on that specifies whether the fan is on (default false).

A double data field named radius that specifies the radius of the fan (default 5).

A string data field named color that specifies the color of the fan (default blue).

A no-arg constructor that creates a default fan.

A method named toString() that returns a string description for the fan.

If the fan is on, the method returns the fan speed, color, and radius in one combined string.

If the fan is not on, the method returns fan color and radius along with the string "fan is off" in one combined string.

Output the fans that are on.

Code –

```
import java.util.Scanner;
import java.util.ArrayList;


public class Fan {
   public static final int SLOW = 1;
   public static final int MEDIUM = 2;
   public static final int FAST = 3;


   private int s = SLOW;
   private boolean o = false;
   private double r = 5;
   private String c = "blue";


   public Fan() {
   }


   public boolean isOn() {
      return o;
   }


   public void setOn(boolean o) {
      this.o = o;
   }


   public String toString() {
      if (o) {
         return "Fan speed: " + s + ", color: " + c + ", radius: " + r;
      } else {
         return "Fan color: " + c + ", radius: " + r + ", fan is off";
      }
   }


   public static void main(String[] args) {
      Scanner scan = new Scanner(System.in);
      ArrayList<Fan> fans = new ArrayList<>();
      fans.add(new Fan());
      int currentFan = 0;
      int ch = 0;


      while (ch != 4) {
         System.out.println("\n===== Fan Control Menu =====");
         System.out.println("1. Turn fan on/off");
         System.out.println("2. Create new fan");
         System.out.println("3. Display running fans");
         System.out.println("4. Exit");
         System.out.print("Enter your choice: ");
```

```
            ch = scan.nextInt();
            scan.nextLine();

            if (ch == 1) {
                fans.get(currentFan).setOn(!fans.get(currentFan).isOn());
                System.out.println("Fan is now " + (fans.get(currentFan).isOn() ? "on" : "off"));
                System.out.println(fans.get(currentFan));
            } else if (ch == 2) {
                fans.add(new Fan());
                currentFan = fans.size() - 1;
                System.out.println("New fan created with default settings (Fan #" + (currentFan + 1) + ")");
                System.out.println(fans.get(currentFan));
            } else if (ch == 3) {
                System.out.println("\n===== Running Fans =====");
                boolean anyRunning = false;
                for (int i = 0; i < fans.size(); i++) {
                    if (fans.get(i).isOn()) {
                        System.out.println("Fan #" + (i + 1) + ": " + fans.get(i));
                        anyRunning = true;
                    }
                }
                if (!anyRunning) {
                    System.out.println("No fans are currently running.");
                }
            } else if (ch == 4) {
                System.out.println("Exiting program");
            } else {
                System.out.println("Invalid choice! Please try again.");
            }
        }
        scan.close();
    }
}
```

OUTPUT –

```
===== Fan Control Menu =====
1. Turn fan on/off
2. Create new fan
3. Display running fans
4. Exit
Enter your choice: 1
Fan is now on
Fan speed: 1, color: blue, radius: 5.0

===== Fan Control Menu =====
1. Turn fan on/off
2. Create new fan
3. Display running fans
4. Exit
Enter your choice: 2
New fan created with default settings (Fan #2)
Fan color: blue, radius: 5.0, fan is off

===== Fan Control Menu =====
1. Turn fan on/off
2. Create new fan
3. Display running fans
4. Exit
Enter your choice: 1
Fan is now on
Fan speed: 1, color: blue, radius: 5.0

===== Fan Control Menu =====
1. Turn fan on/off
2. Create new fan
3. Display running fans
4. Exit
Enter your choice: 3

===== Running Fans =====
Fan #1: Fan speed: 1, color: blue, radius: 5.0
Fan #2: Fan speed: 1, color: blue, radius: 5.0
```

2. Design a class named Account that contains:

An int data field named id for the account (default 0).

A double data field named balance for the account (default 0).

A double data field named annualInterestRate that stores the current interest rate (default 0).

A Date data field named dateCreated that stores the date when the account was created.

A no-arg constructor that creates a default account.

A method named getMonthlyInterestRate() that returns the monthly interest rate.

A method named withDraw that withdraws a specified amount from the account.

A method named deposit that deposits a specified amount to the account.

Create multiple objects and display the account with highest balance.

CODE –

```
import java.util.Date;
import java.util.Scanner;
import
java.text.SimpleDateFormat;
import java.text.ParseException;


public class Account {
   private int id;
   private double money;
   private static double rate; //
Static rate for all accounts
   private Date date;


   public Account() {
      id = 0;
      money = 0;
      date = new Date();
   }


   public Account(int a, double b,
Date d) {
      id = a;
      money = b;
      date = d;
   }


   public int getId() {
      return id;
   }


   public void setId(int a) {
      id = a;
   }


   public double getBalance() {
      return money;
   }


   public void setBalance(double a)
{
      money = a;
   }


   public static double
getAnnualInterestRate() {
      return rate;
   }


   public static void
setAnnualInterestRate(double a) {
      rate = a;
   }


   public Date getDateCreated() {
      return date;
   }


   public double
getMonthlyInterestRate() {
      return rate / 12;
   }


   public double
getMonthlyInterest() {
```

```
        return money * getMonthlyInterestRate() / 100;
    }

    public void withdraw(double a) {
        if (a <= money) {
            money = money - a;
        } else {
            System.out.println("Not enough money");
        }
    }

    public void deposit(double a) {
        if (a > 0) {
            money = money + a;
        }
    }

    public String toString() {
        SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");
        return "Account [id=" + id + ", balance=Rs." + money + ", annualInterestRate=" + rate + "%, dateCreated=" + sdf.format(date) + "]";
    }

    public static Account getHighestBalanceAccount(Account[] a) {
        if (a == null || a.length == 0) {
            return null;
        }

        Account max = a[0];
        for (int i = 1; i < a.length; i++) {
            if (a[i].getBalance() > max.getBalance()) {
                max = a[i];
            }
        }
        return max;
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        Account[] accs = new Account[10];
        int count = 0;

        System.out.print("Enter annual interest rate (%) for all accounts: ");
        double globalRate = scan.nextDouble();
        Account.setAnnualInterestRate(globalRate);

        int choice = 0;
        while (choice != 7) {
            System.out.println("\n==== BANKING SYSTEM MENU =====");
            System.out.println("1. Create New Account");
            System.out.println("2. Display All Accounts");
            System.out.println("3. Deposit");
            System.out.println("4. Withdraw");
            System.out.println("5. Check Monthly Interest");
            System.out.println("6. Find Account with Highest Balance");
            System.out.println("7. Exit");
            System.out.print("Enter your choice: ");

            choice = scan.nextInt();

            if (choice == 1) {
                if (count < accs.length) {
                    System.out.print("Enter account ID: ");
                    int id = scan.nextInt();
                    System.out.print("Enter initial balance: Rs.");
                    double bal = scan.nextDouble();

                    scan.nextLine(); // Clear buffer
                    System.out.print("Enter creation date (DD/MM/YYYY): ");
                    String dateStr = scan.nextLine();

                    Date creationDate = new Date(); // Default to current date
                    try {
                        SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");
                        creationDate = sdf.parse(dateStr);
                    } catch (ParseException e) {
                        System.out.println("Invalid date format! Using current date.");
                    }
```

```
            accs[count] = new
Account(id, bal, creationDate);

            System.out.println("Ac
count created successfully!");

            count++;

          } else {

            System.out.println("Ma
ximum account limit reached!");

          }

        } else if (choice == 2) {

          if (count == 0) {

            System.out.println("No
accounts exist yet.");

          } else {

            System.out.println("All
Account Details:");

            for (int i = 0; i < count;
i++) {

              System.out.println(a
ccs[i]);

            }

          }

        } else if (choice == 3) {

          if (count == 0) {

            System.out.println("No
accounts exist yet.");

          } else {

            System.out.print("Ente
r account ID: ");

            int id = scan.nextInt();

            boolean found = false;



            for (int i = 0; i < count;
i++) {

              if (accs[i].getId() ==
id) {

                System.out.print("
Enter deposit amount: Rs.");

                double amt =
scan.nextDouble();

                accs[i].deposit(amt
);

                System.out.println
("Deposit successful! New
balance: Rs." +
accs[i].getBalance());

                found = true;

                break;

              }

            }



            if (!found) {

              System.out.println("
Account not found!");

            }

          }

        } else if (choice == 4) {

          if (count == 0) {

            System.out.println("No
accounts exist yet.");

          } else {

            System.out.print("Ente
r account ID: ");

            int id = scan.nextInt();

            boolean found = false;



            for (int i = 0; i < count;
i++) {

              if (accs[i].getId() ==
id) {

                System.out.print("
Enter withdrawal amount: Rs.");

                double amt =
scan.nextDouble();

                accs[i].withdraw(a
mt);

                System.out.println
("New balance: Rs." +
accs[i].getBalance());

                found = true;

                break;

              }

            }



            if (!found) {

              System.out.println("
Account not found!");

            }

          }

        } else if (choice == 5) {

          if (count == 0) {

            System.out.println("No
accounts exist yet.");

          } else {

            System.out.print("Ente
r account ID: ");

            int id = scan.nextInt();

            boolean found = false;



            for (int i = 0; i < count;
i++) {

              if (accs[i].getId() ==
id) {

                System.out.println
("Monthly interest: Rs." +
accs[i].getMonthlyInterest());

                found = true;

                break;

              }

            }



            if (!found) {

              System.out.println("
Account not found!");

            }

          }

        } else if (choice == 6) {

          if (count == 0) {

            System.out.println("No
accounts exist yet.");
```

```
            } else {
                Account[] active = new Account[count];
                for (int i = 0; i < count; i++) {
                    active[i] = accs[i];
                }
                Account highest = getHighestBalanceAccount(active);
                System.out.println("Account with highest balance: " + highest);
            }
        } else if (choice == 7) {
            System.out.println("Exiting the system. Goodbye!");
        } else {
            System.out.println("Invalid choice! Please try again.");
        }
    }
    scan.close();
  }
}
```

OUTPUT –

```
Enter annual interest rate (%) for all accounts: 11

===== BANKING SYSTEM MENU =====
1. Create New Account
2. Display All Accounts
3. Deposit
4. Withdraw
5. Check Monthly Interest
6. Find Account with Highest Balance
7. Exit
Enter your choice: 1
Enter account ID: 1
Enter initial balance: Rs.4567
Enter creation date (DD/MM/YYYY): 23/04/2013
Account created successfully!

===== BANKING SYSTEM MENU =====
1. Create New Account
2. Display All Accounts
3. Deposit
4. Withdraw
5. Check Monthly Interest
6. Find Account with Highest Balance
7. Exit
Enter your choice: 1
Enter account ID: 2
Enter initial balance: Rs.567
Enter creation date (DD/MM/YYYY): 12/07/2021
Account created successfully!

===== BANKING SYSTEM MENU =====
1. Create New Account
2. Display All Accounts
3. Deposit
4. Withdraw
5. Check Monthly Interest
6. Find Account with Highest Balance
7. Exit
Enter your choice: 1
Enter account ID: 3
Enter initial balance: Rs.5678
Enter creation date (DD/MM/YYYY): 07/06/2005
Account created successfully!

===== BANKING SYSTEM MENU =====
1. Create New Account
2. Display All Accounts
3. Deposit
4. Withdraw
5. Check Monthly Interest
6. Find Account with Highest Balance
7. Exit
Enter your choice: 3
Enter account ID: 3
Enter deposit amount: Rs.5987
Deposit successful! New balance: Rs.11665.0
```

```
===== BANKING SYSTEM MENU =====
1. Create New Account
2. Display All Accounts
3. Deposit
4. Withdraw
5. Check Monthly Interest
6. Find Account with Highest Balance
7. Exit
Enter your choice: 4
Enter account ID: 2
Enter withdrawal amount: Rs.233
New balance: Rs.334.0

===== BANKING SYSTEM MENU =====
1. Create New Account
2. Display All Accounts
3. Deposit
4. Withdraw
5. Check Monthly Interest
6. Find Account with Highest Balance
7. Exit
Enter your choice: 5
Enter account ID: 2
Monthly interest: Rs.3.061666666666666

===== BANKING SYSTEM MENU =====
1. Create New Account
2. Display All Accounts
3. Deposit
===== BANKING SYSTEM MENU =====
1. Create New Account
2. Display All Accounts
3. Deposit
4. Withdraw
5. Check Monthly Interest
6. Find Account with Highest Balance
7. Exit
Enter your choice: 6
Account with highest balance: Account [id=3, balance=Rs.11665.0, annualInterestRate=11.0%, dateCreated=07/06/2005]
```

3. Design a class named Stock that contains:

A string data field named symbol for the stock's symbol.

A string data field named name for the stock's name.

A double data field named previousClosingPrice that stores the stock price for the previous day.

A double data field named currentPrice that stores the stock price for the current time.

A constructor that creates a stock with specified symbol and name.

A method named changePercent() that returns the percentage changed from previousClosingPrice to currentPrice.

Create 10 objects and output the Stocks having the Highest and the Lowest Previous Closing Price.

CODE –

```
import java.util.Scanner;


public class Stock {
   private String sym;
   private String nm;
   private double oldPrice;
   private double newPrice;

   public Stock(String sym, String nm) {
      this.sym = sym;
      this.nm = nm;
   }

   public String getSymbol() {
      return sym;
   }

   public String getName() {
      return nm;
   }

   public double getOldPrice() {
      return oldPrice;
   }

   public void setOldPrice(double old) {
      this.oldPrice = old;
   }

   public double getNewPrice() {
      return newPrice;
   }

   public void setNewPrice(double new1) {
      this.newPrice = new1;
   }

   public double getPercent() {
      return ((newPrice - oldPrice) / oldPrice) * 100;
   }

public String toString() {
   return sym + " - " + nm;
}

   public static void main(String[] args) {
      Scanner scan = new Scanner(System.in);

      Stock[] list = new Stock[10];
      list[0] = new Stock("AAPL", "Apple Inc");
      list[0].setOldPrice(15000.95);
      list[0].setNewPrice(15200.52);

      list[1] = new Stock("MSFT", "Microsoft Corporation");
      list[1].setOldPrice(28000.11);
      list[1].setNewPrice(28200.67);

      list[2] = new Stock("GOOGL", "Alphabet Inc");
      list[2].setOldPrice(10900.86);
      list[2].setNewPrice(11000.20);
```

```
        list[3] = new Stock("AMZN",
"Amazon.com Inc");
        list[3].setOldPrice(10700.33);
        list[3].setNewPrice(10650.95)
;

        list[4] = new Stock("TSLA",
"Tesla Inc");
        list[4].setOldPrice(17800.65);
        list[4].setNewPrice(17950.50)
;

        list[5] = new Stock("META",
"Meta Platforms Inc");
        list[5].setOldPrice(25000.55);
        list[5].setNewPrice(25400.78)
;

        list[6] = new Stock("NVDA",
"NVIDIA Corporation");
        list[6].setOldPrice(36000.20);
        list[6].setNewPrice(36600.35)
;

        list[7] = new Stock("JPM",
"JPMorgan Chase & Co");
        list[7].setOldPrice(12100.43);
        list[7].setNewPrice(12200.10)
;

        list[8] = new Stock("V", "Visa
Inc");
        list[8].setOldPrice(20050.10);
        list[8].setNewPrice(19950.82)
;

        list[9] = new Stock("WMT",
"Walmart Inc");
        list[9].setOldPrice(4950.84);
        list[9].setNewPrice(4975.12);

        int opt;
        do {
            System.out.println("\n====
= Stock Menu =====");
            System.out.println("1.
Show All Stocks");
            System.out.println("2. Find
Highest Stock");
            System.out.println("3. Find
Lowest Stock");
            System.out.println("4.
Check Price Change");
            System.out.println("5.
Exit");
            System.out.print("Choose
option: ");
            opt = scan.nextInt();

            if (opt == 1) {
                showAll(list);
            } else if (opt == 2) {
                findMax(list);
            } else if (opt == 3) {
                findMin(list);
            } else if (opt == 4) {
                checkChange(list, scan);
            } else if (opt == 5) {
                System.out.println("Bye!"
);
            } else {
                System.out.println("Wron
g input!");
            }
        } while (opt != 5);

        scan.close();
    }

    private static void
showAll(Stock[] list) {
        System.out.println("\n=====
All Stocks =====");
        for (int i = 0; i < list.length;
i++) {
            System.out.println(list[i] + "
(Old: Rs" + list[i].getOldPrice() +
                    ", New: Rs" +
list[i].getNewPrice() + ")");
        }
    }

    private static void
findMax(Stock[] list) {
        Stock max = list[0];
        for (int i = 1; i < list.length;
i++) {
            if (list[i].getOldPrice() >
max.getOldPrice()) {
                max = list[i];
            }
        }
        System.out.println("\nHighest
Stock: " + max + " (Rs" +
max.getOldPrice() + ")");
    }

    private static void
findMin(Stock[] list) {
        Stock min = list[0];
        for (int i = 1; i < list.length;
i++) {
            if (list[i].getOldPrice() <
min.getOldPrice()) {
                min = list[i];
            }
        }
```

```
        System.out.println("\nLowest Stock: " + min + " (Rs" + min.getOldPrice() + ")");
    }

    private static void checkChange(Stock[] list, Scanner scan) {
        System.out.println("\n===== Stocks =====");
        for (int i = 0; i < list.length; i++) {
            System.out.println((i+1) + ". " + list[i]);
        }
        System.out.print("Pick a number (1-10): ");
        int num = scan.nextInt() - 1;

        if (num >= 0 && num < list.length) {
            Stock pick = list[num];
            double change = pick.getPercent();
            System.out.println("\nStock: " + pick);
            System.out.println("Old Price: Rs" + pick.getOldPrice());
            System.out.println("New Price: Rs" + pick.getNewPrice());
            System.out.println("Change: " + change + "%");
            if (change > 0) {
                System.out.println("Status: UP ");
            } else if (change < 0) {
                System.out.println("Status: DOWN ");
            } else {
                System.out.println("Status: SAME ");
            }
        } else {
            System.out.println("Wrong number!");
        }}}
```

OUTPUT –

```
===== Stock Menu =====
1. Show All Stocks
2. Find Highest Stock
3. Find Lowest Stock
4. Check Price Change
5. Exit
Choose option: 1

===== All Stocks =====
AAPL - Apple Inc (Old: Rs15000.95, New: Rs15200.52)
MSFT - Microsoft Corporation (Old: Rs28000.11, New: Rs28200.67)
GOOGL - Alphabet Inc (Old: Rs10900.86, New: Rs11000.2)
AMZN - Amazon.com Inc (Old: Rs10700.33, New: Rs10650.95)
TSLA - Tesla Inc (Old: Rs17800.65, New: Rs17950.5)
META - Meta Platforms Inc (Old: Rs25000.55, New: Rs25400.78)
NVDA - NVIDIA Corporation (Old: Rs36000.2, New: Rs36600.35)
JPM - JPMorgan Chase & Co (Old: Rs12100.43, New: Rs12200.1)
V - Visa Inc (Old: Rs20050.1, New: Rs19950.82)
WMT - Walmart Inc (Old: Rs4950.84, New: Rs4975.12)

===== Stock Menu =====
1. Show All Stocks
2. Find Highest Stock
3. Find Lowest Stock
4. Check Price Change
5. Exit
Choose option: 2

Highest Stock: NVDA - NVIDIA Corporation (Rs36000.2)

===== Stock Menu =====
1. Show All Stocks
2. Find Highest Stock
3. Find Lowest Stock
4. Check Price Change
5. Exit
Choose option: 3

Lowest Stock: WMT - Walmart Inc (Rs4950.84)

===== Stock Menu =====
1. Show All Stocks
2. Find Highest Stock
3. Find Lowest Stock
4. Check Price Change
5. Exit
Choose option: 4

===== Stocks =====
1. AAPL - Apple Inc
2. MSFT - Microsoft Corporation
3. GOOGL - Alphabet Inc
4. AMZN - Amazon.com Inc
5. TSLA - Tesla Inc
6. META - Meta Platforms Inc
7. NVDA - NVIDIA Corporation
8. JPM - JPMorgan Chase & Co
9. V - Visa Inc
10. WMT - Walmart Inc
Pick a number (1-10): 3

Stock: GOOGL - Alphabet Inc
Old Price: Rs10900.86
New Price: Rs11000.2
Change: 0.9113042457200636%
Status: UP
```

4. A Date class models a calendar date with day, month, and year.

It contains the following members:

3 private instance variables: day, month, and year.

a. Constructors, public getters and setters for the private instance variables.

b. A method setDate(), which sets the day, month and year.

c. A toString(), which returns "DD/MM/YYYY", with leading zero for DD and MM if applicable.

Write the Date class and a test driver to test all the public methods. No input validations are required for day, month, and year.

CODE –

```
public class date {

   private int day;

   private int month;

   private int year;


   public date() {

      this.day = 1;

      this.month = 1;

      this.year = 2023;

   }


   public date(int day, int month,
int year) {

      this.day = day;

      this.month = month;

      this.year = year;

   }


   public int getDay() {

      return day;

   }


   public int getMonth() {

      return month;

   }

   public int getYear() {

      return year;

   }


   public void setDay(int day) {

      this.day = day;

   }


   public void setMonth(int
month) {

      this.month = month;

   }


   public void setYear(int year) {

      this.year = year;

   }


   public void setDate(int day, int
month, int year) {

      this.day = day;

      this.month = month;

      this.year = year;

   }

   @Override

   public String toString() {

      return
String.format("%02d/%02d/%04d
", day, month, year);

   }


   public static void main(String[]
args) {

      java.util.Scanner scanner =
new java.util.Scanner(System.in);

      date myDate = new date();

      int choice;


      do {

         System.out.println("\n-----
Date Operations Menu -----");

         System.out.println("1.
Create a new date");

         System.out.println("2.
Display date");

         System.out.println("3.
Exit");

         System.out.print("Enter
your choice: ");


         choice = scanner.nextInt();
```

```
        switch (choice) {

            case 1:

                System.out.print("Ente
r day: ");

                int day =
scanner.nextInt();

                System.out.print("Ente
r month: ");

                int month =
scanner.nextInt();

                System.out.print("Ente
r year: ");

                int year =
scanner.nextInt();

                myDate = new
date(day, month, year);

                System.out.println("N
ew date created: " + myDate);

                break;


            case 2:

                System.out.println("Cu
rrent date: " + myDate);

                break;


            case 3:

                System.out.println("Ex
iting program. Goodbye!");

                break;


            default:

                System.out.println("In
valid choice. Please try again.");

        }


    } while (choice != 3);


    scanner.close();

  }

}
```

OUTPUT –

```
----- Date Operations Menu -----
1. Create a new date
2. Display date
3. Exit
Enter your choice: 1
Enter day: 2
Enter month: 3
Enter year: 2019
New date created: 02/03/2019

----- Date Operations Menu -----
1. Create a new date
2. Display date
3. Exit
Enter your choice: 2
Current date: 02/03/2019
```

5. A Point class models a 2D point at (x, y), as shown in the class diagram. It contains the following members:

2 private instance variables x and y, which maintain the location of the point.

Constructors, getters and setters.

A method setXY(), which sets the x and y of the point; and a method getXY(), which returns the x and y in a 2-element int array.

A toString(), which returns "(x,y)".

3 versions of overloaded distance():

o distance(int x, int y) returns the distance from this instance to the given point at (x, y).

o distance(Point another) returns the distance from this instance to the given Point instance (called another).

o distance() returns the distance from this instance to (0,0).

CODE –

```
public class point {
   private int x;
   private int y;

   public point() {
      this.x = 0;
      this.y = 0;
   }

   public point(int x, int y) {
      this.x = x;
      this.y = y;
   }

   public int getX() {
      return x;
   }

   public void setX(int x) {
      this.x = x;
   }

   public int getY() {
      return y;
   }

   public void setY(int y) {
      this.y = y;
   }

   public void setXY(int x, int y) {
      this.x = x;
      this.y = y;
   }

   public int[] getXY() {
      return new int[]{x, y};
   }

   @Override
   public String toString() {
      return "(" + x + "," + y + ")";
   }

   public double distance(int x, int y) {
      int xDiff = this.x - x;
      int yDiff = this.y - y;
      return Math.sqrt(xDiff * xDiff + yDiff * yDiff);
   }

   public double distance(point another) {
      return distance(another.getX(), another.getY());
   }

   public double distance() {
      return distance(0, 0);
   }

   public static void main(String[] args) {
      java.util.Scanner scanner = new java.util.Scanner(System.in);
      point p = new point(0, 0);
      int choice;

      do {
         System.out.println("\n---- Point Operations Menu ----");
         System.out.println("1. Set point coordinates (x,y)");
         System.out.println("2. Get point coordinates");
         System.out.println("3. Calculate distance to origin (0,0)");
         System.out.println("4. Calculate distance to specific point");
         System.out.println("5. Calculate distance to another Point object");
         System.out.println("0. Exit");
         System.out.print("Enter your choice: ");

         choice = scanner.nextInt();

         switch (choice) {
            case 1:
```

```
                System.out.print("Ente
r x coordinate: ");

                int x =
scanner.nextInt();

                System.out.print("Ente
r y coordinate: ");

                int y =
scanner.nextInt();

                p.setXY(x, y);

                System.out.println("Poi
nt set to " + p);

                break;


            case 2:

                int[] coordinates =
p.getXY();

                System.out.println("Cu
rrent coordinates: x = " +
coordinates[0] + ", y = " +
coordinates[1]);

                break;

            case 3:

                System.out.println("Dis
tance to origin (0,0): " +
p.distance());

                break;

            case 4:

                System.out.print("Ente
r x coordinate of target point: ");

                x = scanner.nextInt();

                System.out.print("Ente
r y coordinate of target point: ");

                y = scanner.nextInt();

                System.out.println("Dis
tance to point (" + x + "," + y + "): "
+ p.distance(x, y));

                break;

            case 5:

                System.out.print("Ente
r x coordinate of another point: ");

                x = scanner.nextInt();

                System.out.print("Ente
r y coordinate of another point: ");

                y = scanner.nextInt();

                point another = new
point(x, y);

                System.out.println("Dis
tance to point " + another + ": " +
p.distance(another));

                break;

            case 0:

                System.out.println("Exi
ting program. Goodbye!");

                break;

            default:

                System.out.println("Inv
alid choice. Please try again.");

        }

    } while (choice != 0);

    scanner.close();

  }
```

OUTPUT -

```
---- Point Operations Menu ----
1. Set point coordinates (x,y)
2. Get point coordinates
3. Calculate distance to origin (0,0)
4. Calculate distance to specific point
5. Calculate distance to another Point object
0. Exit
Enter your choice: 1
Enter x coordinate: 4
Enter y coordinate: 7
Point set to (4,7)

---- Point Operations Menu ----
1. Set point coordinates (x,y)
2. Get point coordinates
3. Calculate distance to origin (0,0)
4. Calculate distance to specific point
5. Calculate distance to another Point object
0. Exit
Enter your choice: 2
Current coordinates: x = 4, y = 7

---- Point Operations Menu ----
1. Set point coordinates (x,y)
2. Get point coordinates
3. Calculate distance to origin (0,0)
4. Calculate distance to specific point
5. Calculate distance to another Point object
0. Exit
Enter your choice: 3
Distance to origin (0,0): 8.062257748298550

---- Point Operations Menu ----
1. Set point coordinates (x,y)
2. Get point coordinates
3. Calculate distance to origin (0,0)
4. Calculate distance to specific point
5. Calculate distance to another Point object
0. Exit
Enter your choice: 4
Enter x coordinate of target point: 5
Enter y coordinate of target point: 8
Distance to point (5,8): 1.4142135623730951

---- Point Operations Menu ----
1. Set point coordinates (x,y)
2. Get point coordinates
3. Calculate distance to origin (0,0)
4. Calculate distance to specific point
5. Calculate distance to another Point object
0. Exit
Enter your choice: 5
Enter x coordinate of another point: 7
Enter y coordinate of another point: 8
Distance to point (7,8): 3.1622776601683795
```

6. A class called Time, which models a time instance with hour, minute and second, is designed as shown in the class diagram. It contains the following members:

3 private instance variables hour, minute, and second.

Constructors, getters and setters.

A method setTime() to set hour, minute and second.

A toString() that returns "hh:mm:ss" with leading zero if applicable.

A method nextSecond() that advances this instance by one second.

It returns this instance to support chaining (cascading) operations, e.g., t1.nextSecond().nextSecond().

Take note that the nextSecond() of 23:59:59 is 00:00:00.

Write the Time class and a test driver to test all the public methods. No input validations are required.

CODE –

```
public class time {
   private int hour;
   private int minute;
   private int second;
   public time() {
      this.hour = 0;
      this.minute = 0;
      this.second = 0;
   }

   public time(int hour, int
minute, int second) {
      this.hour = hour;
      this.minute = minute;
      this.second = second;
   }
   public int getHour() {
      return hour;
   }

   public int getMinute() {
      return minute;
   }
   public int getSecond() {
      return second;
   }

   public void setHour(int hour) {
      this.hour = hour;
   }
   public void setMinute(int
minute) {
      this.minute = minute;
   }
   public void setSecond(int
second) {
      this.second = second;
   }

   public void setTime(int hour,
int minute, int second) {
      this.hour = hour;
      this.minute = minute;
      this.second = second;
   }
public time nextSecond() {
   second++;
   if (second >= 60) {
      second = 0;
      minute++;
      if (minute >= 60) {
         minute = 0;
         hour++;
         if (hour >= 24) {
            hour = 0;
         }
      }
   }
   return this;
}
@Override
public String toString() {
```

```
    return
String.format("%02d:%02d:%02d
", hour, minute, second);

  }

  public static void main(String[]
args) {

    java.util.Scanner scanner =
new java.util.Scanner(System.in);

    time t = new time(12, 0, 0);

    int choice;



    do {

      System.out.println("\nCurr
ent Time: " + t);

      System.out.println("Menu:
");

      System.out.println("1. Set
Time (Hour, Minute, Second)");

      System.out.println("2.
Next Second");

      System.out.println("3. Get
Hour, Minute, Second");

      System.out.println("0.
Exit");

      System.out.print("Enter
your choice: ");



      choice = scanner.nextInt();

      switch (choice) {

        case 1:

          System.out.print("Ente
r hour (0-23): ");

          int h =
scanner.nextInt();

          System.out.print("Ente
r minute (0-59): ");

          int m =
scanner.nextInt();

          System.out.print("Ente
r second (0-59): ");

          int s =
scanner.nextInt();

          t.setTime(h, m, s);

          break;

        case 2:

          t.nextSecond();

          System.out.println("A
dvanced to next second!");

          break;

        case 3:

          System.out.println("H
our: " + t.getHour());

          System.out.println("M
inute: " + t.getMinute());

          System.out.println("Se
cond: " + t.getSecond());

          break;

        case 0:

          System.out.println("Ex
iting program...");

          break;

        default:

          System.out.println("In
valid choice. Please try again.");

      }

    } while (choice != 0);


    scanner.close();

  }

}
```

OUTPUT –

```
Menu:
1. Set Time (Hour, Minute, Second)
2. Next Second
3. Get Hour, Minute, Second
0. Exit
Enter your choice: 1
Enter hour (0-23): 23
Enter minute (0-59): 59
Enter second (0-59): 59

Current Time: 23:59:59
Menu:
1. Set Time (Hour, Minute, Second)
2. Next Second
3. Get Hour, Minute, Second
0. Exit
Enter your choice: 3
Hour: 23
Minute: 59
Second: 59

Current Time: 23:59:59
Menu:
1. Set Time (Hour, Minute, Second)
2. Next Second
3. Get Hour, Minute, Second
0. Exit
Enter your choice: 2
Advanced to next second!
```