SAMPLE

CSE21

SAMPLE • Test 1 • VersionA
Time: 50 minutes
Maximum Points: 200

Name : SOLUTION

The following precedence table is provided for your use:

Precedence of Operators
( )
- (unary), !, ++, --
*, /, %
+ , - (binary)
<, <=, >, >=
= =, !=
&&
‖
=, +=, -=, *=, /=, %=

Otherwise left to right

Write all answers in the boxes or on the lines provided.

1. (40 points) Suppose SetPayment is a polymorphic (overloaded) function. The start of the SetPayment functions are shown below; assume there is sensible code within the curley-brackets.

```
1.        int setPayment(int) { … }

2.        int setPayment(int[]) { … }

3.        int setPayment(double) { … }

4.        int setPayment(int, int) { … }

5.        int setPayment(int, double) { … }

6.        int setPayment(double, double) { … }

7.        int setPayment(int, double, int) { … }

8.        int setPayment(int, int, double) { … }

9.        int setPayment(int, int, int) { … }

10.       int setPayment(int, int, double[]) { … }

11.       No corresponding method definition
```

Put the corresponding method above for the call below.

```
                        double y=299.98,z=567.88;
                        int w = 5, t=60, n = 15;
                        double[] darr = new double[10];
                        int[] iarr = new int[10];

_____5______      x = setPayment(w,y);

_____7______      y = setPayment(10,z,t);

_____11______     z = setPayment(y,t,15);

_____2______      x = setPayment(iarr);

_____6______      t = setPayment(y, z);

_____3______      x = setPayment(5.0);

_____11______     n = setPayment(w, 2.0, y);

_____4______      x = setPayment(w, iarr[w]);

_____1,3______    x = setPayment(setPayment(darr[2]));

_____10______     x = setPayment(2,t,darr);
```

2. (a) (20 pts) Give the output of the following program fragment:

```
int [] arr = {1, 2, 3, 4, 2, 3};
int x = 2;
for (int i = 0; i < arr.length; i++) {
    if (arr[i] == x) {
        System.out.println(i);
    }
 }
```

```
1
4
```

(b) (40 pts) Write an iterative method that receives as argument an array of integers and prints the square of each element, one at each line. For example, if the array had contents from (a) it should print out:

```
1
4
9
16
4
9
```

The method will also keep track of the biggest squared element and returns that to the caller. Again with the example array from (a) the return value would be 16. If the input array is empty (has no values) then it should return 0.

```
static int squarePrint(int[] arr) {
      int max = 0, squared = 0;;
      for (int i = 0; i < arr.length; i++) {
            System.out.println(squared = arr[i]*arr[i]);
            if (squared > max)
                  max = squared;
      }
      return max;
}
```

3. (30 points) The following program should compute the volume of a sphere. The formula is shown below.

sphere = (4/3) x Π x **r** x **r** x **r**

Add a method named `CalcVolume` to the following class. This function should receive two arguments, the value of `pi` and `radius` of the sphere. The function should then calculate the `volume` of the sphere and return the `volume` to `main`. Write your calculation to accommodate. Add the code as well and a call from `main` to the new function.

```
public class Sphere {

      public static double calcVolume(double r, double p) {

            return (4.0/3.0*p * r * r * r);

      }
```

Write the method `calcVolume.` This function should receive two arguments, `r` and `p`. Calculate the volume according to the formula above and return the volume.

```
  public static void main(String[] args) {
      double pi, radius,   volume;
      Scanner kbd = new Scanner(System.in);
      System.out.println("Please enter the radius of the sphere:");
      radius = kbd.nextDouble();
      System.out.println("Please enter an approximation for Pi:");
      pi = kbd.nextDouble( );

      volume = calcVolume(radius, pi);

      System.out.println("The volume of the sphere is: "+ volume);
}
```

Add a call to the function `CalcVolume.` Send the `radius` and `pi` as arguments and return the `volume.` Store the volume in the variable named `volume`.

4. (40 points) Do the following for the `SalesRecord` class specification shown below. Use the back of the previous page as extra space if needed.

   a) Write a method named `initialize`. This method takes in a double array pointer as a parameter and initializes each value in the array to `100.0`.

   b) Write a `setBonus` method. The `setBonus` method does not return a value, and should receive three parameters: array pointer to bonuses, an integer representing which bonus to set (0 through 11), and a double representing the bonus for that month. Thus, if an employee should receive a $1575.50 bonus in January, you would call `setBonus(bonus, 0, 1575.50)`. The method should also perform error checking, to make sure the index number is in the correct range and the bonus amount is positive, before placing the amount into the array.

   c) Write a `calcSalary` (bonus, monthlySalary) method. The `calcSalary` method should return a number representing the total salary for the year. This is calculated as follows: multiply the monthly salary amount by 12, and add to that the total of all the bonus amounts in the bonus array. Return this amount as a double.

```
// File: SalesRecord.java  Java source code for SalesRecord class specification.
import java.util.*;

public class SalesRecord {
```

```
// a)
        public static void initialize(double[] arr) {
                for (int i = 0; i < arr.length; i++)
                    arr[i] = 100.0;
        }
```

```
// b)

public static void setBonus(double[] bonus, int index, double amount) {
        if (index >= 0 && index < 12 && amount >= 0.0)
                bonus[index] = amount;
        else
                System.out.println("Invalid entry");  // not required
}
```

```
// c)

public static double calcSalary(double[] bonus, double monthlySalary ){

        double totalBonus = 0.0;
        for (int i=0; i < bonus.length; i++)
                totalBonus += bonus[i];
        return monthlySalary*12 + totalBonus;

}
```

5. (30 points) Use the information in the `SalesRecord` class on the previous page to write the Java statements described below in the following program segment.

```
public static main(String[] args) {
   double totalSalary;
   Scanner kbd = new Scanner(System.in);

   String name;            // employee's name
   int year;                // calendar year
   double monthlySalary;    // employee's monthly salary without bonus
   double[] bonus;         // employee's bonuses for this year
```

```
   System.out.print("Please enter name: ");
   name = kbd.next();
```

Ask user for employee’s name and set it to the variable *name*.

```
   System.out.print("Please enter year: ");
   year = kbd.nextInt();
```

Ask user for year and set it to the variable *year*.

```
   System.out.print("Please enter monthly Salary: ");
   monthlySalary = kbd.nextDouble();
```

Ask user for employee’s monthly salary and set it to the variable *monthlySalary*.

```
   bonus = new double[12];
   initialize(bonus);
```

Allocate 12 spaces for array *bonus* and call initialize with it.

```
   for (int i = 0; i < bonus.length; i++) {
     System.out.print("Please enter " + i + "month bonus:
         ");
     double amount = kdb.nextDouble();
     if (amount > 100)
       setBonus(bonus, i, amount);
   }
```

Write a loop to ask user for each month’s bonus and if the value is above 100 then set it to the new value by calling *setBonus*.

```
   totalSalary = calcSalary(bonus, monthlySalary);
   System.out.println("Total salary is " + totalSalary);
```

Call *calcSalary* and store the returned value in the variable `totalSalary`. Then print out that value on the screen.

```
}
```