## Question 1: (18 Marks)

Determine the output for each of the following code snippets (assuming successful compilation):

**a) (**3 Marks**)**

```
for (int i = 0; i < 5; i++) {
    for (int j = 0; j <= 5; j++) {
        if (j == i + 1) break;
        System.out.print(j);
    }
    System.out.println();
}
```

**b) (**3 Marks**)**

```
for (int i = 0; i < 5; i++) {
    for (int j = 0; j <= 5; j++) {
        if (j == i) continue;
        System.out.print(j);
    }
    System.out.println();
}
```

**c) (**3 Marks**)**

```
int i = 0;
do {
    System.out.println(++i);
    System.out.println(i++);
} while (i < 5);
```

**d) (**3 Marks**)**

```
for (int i = 0; i < 5; i++) {
    for (int j = 0; j < 5; j++)
        System.out.print((i+j)%2==0?"■":" ");
    System.out.println();
}
```

**e) (**3 Marks**)**

```
for (int i = 0; i < 5; i++) {
    for (int j = 0; j < 5; j++) {
        char c;
        switch (Math.abs(i - j)) {
            case 0 : c = '\\'; break;
            case 2 : c = '+' ; break;
            case 4 : c = '.' ; break;
            default: c = ' ' ; break;
        }
        System.out.print(c);
    }
    System.out.println();
}
```

**f) (**3 Marks**)**

```
{
    int f1 = 1;
    int f2 = 1;
    for (int i = 1; i <= 5; i += 1) {
        System.out.println(f1);
        f2 = f1 + f2;
        f1 = f2 - f1;
    }
}
```

## Solution:

**a)**

```
0
01
012
0123
01234
```

**b)**

```
12345
02345
01345
01245
01235
```

**c)**

```
1
1
3
3
5
5
```

**d)**

**e)**

```
\ + .
 \ +
+ \ +
 + \
. + \
```

**f)**

```
1
1
2
3
5
```

## Question 2: (18 Marks)

For the three questions that you will solve later:

**a)** (6 Marks)
avoid syntax and runtime errors,

**b)** (6 Marks)
validate the user input,

**c)** (6 Marks)
prompt the user with meaningful instructions, and

**d)** (6 Marks)[$]
write the code using a clean style.

### Solution:

For example, the student should

**a)**
import necessary packages,
declare any used variables,
write every statement in the right syntax,

**b)**
check that every value entered by the user falls within the acceptable range,

**c)**
print a meaningful hint before reading any input from the user,
print a meaningful label before printing any output to the user,
print a meaningful message whenever something wrong happens,

**d)**
use indentation,
follow naming conventions,
align braces, and
write comments.

$ Bonus

## Question 3: (18 Marks)

Write a full program including three methods for printing the following patterns using only one '*' and one ' ' per method.

**a)** (6 Marks)

```
*****
*   *
* * *
*   *
*****
```

**b)** (6 Marks)

```
*   *
 * *
  *
 * *
*   *
```

**c)** (6 Marks)

```
  *
 ***
*****
 ***
  *
```

## Solution:

```
public class Q_3 {

    public static void main(String args[]) {
        {
            System.out.println("a)");
            q_3_a();
            System.out.println("b)");
            q_3_b();
            System.out.println("c)");
            q_3_c();
        }
    }

//Function Definition
//a)
    static void q_3_a() {
        for (int i = 0; i < 5; i++) {
            for (int j = 0; j < 5; j++) {
                System.out.print((i == 0 || i == 4 || j == 0 || j == 4 || (i == 2 && j == 2)) ? '*' : ' ');
            }
            System.out.println();
        }
    }
//b)
    static void q_3_b() {
        for (int i = 0; i < 5; i++) {
            for (int j = 0; j < 5; j++) {
                System.out.print((i - j == 0 || i + j == 4) ? '*' : ' ');
            }
            System.out.println();
        }
    }
//c)
    static void q_3_c() {
        for (int i = 0; i < 5; i++) {
            for (int j = 0; j < 5; j++) {
                System.out.print((Math.abs(j - 2) + Math.abs(i - 2) <= 2) ? '*' : ' ');
            }
            System.out.println();
        }
    }
}
```

## Question 4: (18 Marks)

The factorial of a nonnegative integer $n$ is written as $n!$ (pronounced "n factorial") and is defined as follows:

$$n! = \begin{cases} n \cdot (n-1) \cdot (n-2) \cdot \ldots \cdot 1 & , n \geqslant 1 \\ 1 & , n = 0 \end{cases}$$

**For example**, $5! = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$, which is $120$.

Write a full program including three methods:

**a)** (6 Marks)

`fact` that takes a nonnegative integer and returns its factorial,

**b)** (6 Marks)

`nbase` that estimates the value of the mathematical constant $e$ by using the following formula, and $e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \ldots + \frac{1}{n!}$.

**c)** (6 Marks)

`nexp` that takes a real number $x$ and computes the value of $e^x$ by using the following formula.

$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots + \frac{x^n}{n!}$.

## Solution:

```
public class Q_4 {

    public static void main(String args[]) {
        {
            System.out.println(fact(4));
            System.out.println(fact(-4));
            System.out.println(nbase(10));
            System.out.println(nexp(2, 10));
        }
    }

//a)
    static int fact(int n) {
        int f;
        if (n >= 0) {
            f = 1;
            for (int i = 1; i <= n; i++) {
                f *= i;
            }
        } else {
            f = -1;
        }
        return f;
    }
//b)
    static double nbase(int n) {
        long f = 1;
        double e = 1;
        for (int i = 1; i <= n; i++) {
            f *= i;
            e += 1.0 / f;
        }
        return e;
    }
//c)
    static double nexp(int x, int n) {
        long f = 1;
        double ex = 1, exp = 1.0;
        for (int i = 1; i <= n; i++) {
            f *= i;
            exp *= x;
            ex += exp / f;
        }
        return ex;
    }
}
```

## Question 5: (18 Marks)

In survey engineering, a *traverse* is an $n$-sided closed polygon. *Traverse angle balancing* is a process intended for adjusting (correcting) the measured internal angles of a given traverse according to the following equations. The target of this process is to make the actual sum of the corrected angles the same as the theoretical sum ($tsum$).

$$tsum = 180 \cdot (n-2)$$
$$asum = \left(\sum_{i=1}^{n} a_i\right)$$
$$error = asum - tsum$$
$$correction = error/n$$
$$\hat{a}_i = a_i - correction \forall i \in [1, n]$$

Create a class `Traverse` and provide:

**a)** (6 Marks)
a constructor that takes an array with three or more traverse angles $a_1, a_2, \ldots, a_n$,

**b)** (6 Marks)
a method `correct` for performing *traverse angle balancing*, and

**c)** (6 Marks)
a method `getAngles` that returns the traverse angles.

**Example**: If the measured angles are $a = \{61.5, 60.5, 59.5\}$, then the corrected angles should be $\hat{a} = \{61.0, 60.0, 59.0\}$

## Solution:

```
public class Q_5 {

    public static void main(String args[]) {
        {
            Traverse t = new Traverse(new double[]{61.5, 60.5, 59.5});
            t.correct();
            double a[] = t.getAngles();
            System.out.println("Corrected angles:");
            for (int i = 0; i < a.length; i++) {
                System.out.println(a[i]);
            }
        }
    }
}

public class Traverse {

    protected double[] a;
//a)
    public Traverse(double[] a) {
        this.a = a;
    }
//b)
    public void correct() {
        int n = a.length;
        double tsum, sum = 0, error, correction;
        //Calculate actual and theoretical sums
        for (int i = 0; i <= n - 1; i += 1) {
            sum = sum + a[i];
        }
        tsum = 180 * (n - 2);
        //Calculate error and correction
        error = sum - tsum;
        correction = error / n;
        //Correct angles and print results
        for (int i = 0; i < a.length; i += 1) {
            a[i] = a[i] - correction;
        }
    }
//c)
    public double[] getAngles() {
        return a;
    }
}
```

## Question 6: (18 Marks)

An $n^{th}$ degree polynomial is expressed as:

$$f(x) = \sum_{i=0}^{n} a_i x^i, a_n \neq 0$$

Create a class `Polynomial` and provide:

**a)** (6 Marks)
a constructor that takes an array to initialize the polynomial parameters.

**b)** (6 Marks)
a method `f` that takes a real number $x$ and returns the value of $f(x)$.

**c)** (6 Marks)
a method `toString` that returns a string representing the polynomial on the form:
$f(x) = a_0 + a_1 x + \ldots + a_n x^n$

**Example**: Assuming that the polynomial parameters are {1, 0, -2, 3}, the method `f(1)` should return 2.0 and the method `toString()` should return:

```
f(x) = 1.00 - 2.00x^2 + 3.00x^3
```

## Solution:

```
public class Q_6 {

    public static void main(String args[]) {
        {
            Polynomial p = new Polynomial(new double[]{1, 0, -2, 3});
            System.out.println(p.f(1));
            System.out.println("f(x) = " + p.toString());
        }
    }
}

public class Polynomial {

    protected double[] a;
//a)
    public Polynomial(double[] a) {
        this.a = a;
    }
//b)
    public double f(double x) {
        double ret = 0;
        double prod = 1;
        for (int i = 0; i < a.length; i++) {
            ret += a[i] * prod;
            prod *= x;
        }
        return ret;
    }
//c)
    @Override
    public String toString() {
        String ret = "";
        boolean first = true;
        for (int i = 0; i < a.length; i++) {
            //skip zero terms
            if (a[i] != 0) {
                //+/-
                ret += (a[i] < 0 ? " - " : !first ? " + " : "");
                if (Math.abs(a[i]) != 1 || i == 0) {
                    //a
                    ret += String.format("%.2f", Math.abs(a[i]));
                }
                if (i > 0) {
                    //x
                    ret += "x";
                    if (i > 1) {
                        //^i
                        ret += String.format("^%d", i);
                    }
                }
                first = false;
            }
        }
        return ret;
    }
}
```

Good Luck
Dr. Islam ElShaarawy