Universidad de Costa Rica
Sede del Atlántico

Bach. Informática Empresarial
IF-2000 Programación I

Parcial 2

Carné: _____ **Nombre:** _____

1. [6 pts] Considere el siguiente método recursivo denominado recorreArreglo:

```
public class Parcial2{

    public static void recorreArreglo(int[] arreglo, int i) {
        if (i < arreglo.length) {
            recorreArreglo(arreglo, i + 1);
            System.out.println(arreglo[i]);
        }
    }

    public static void main(String[] args) {

        // Ejemplo de prueba

        int[] vec = {8, 2, 5, 12, 6, 8};
        recorreArreglo(vec, 0);
    }
}
```

a) [3 pts.] Indique la salida que genera el método recorreArreglo para el ejemplo de prueba (vec).

R:// 8, 6, 12, 5, 2, 8

b) [3 pts.] Haga una versión no recursiva del método recorreArreglo (la misma salida).

R://

```
public static void recorreArreglo(int[] arreglo) {
     for (int i= arreglo.length-1;i >= 0;i--){
          System.out.println(arreglo[i]);
     }
}
```

2. [12 pts.] Considere el siguiente código:

```
public class Parcial2{

    public static void recorreArreglo(int[][][] arreglo) {

        for (int f = arreglo[0].length - 1; f >= 0; f--) {
            for (int t = arreglo.length - 1; t >= 0; t--) {
                for (int c = arreglo[0][0].length - 1; c >= 0; c--) {
                        System.out.println(arreglo[t][f][c]);
                }
            }
        }
    }


    public static void main(String[] args) {

        // Ejemplo de prueba
```

```
        int[][][] cubo = {
                            {{1, 2, 3}, {4, 5, 6}},
                            {{7, 8, 9}, {10, 11, 12}},
                            {{13, 14, 15}, {16, 17, 18}}
                          };

        recorreArreglo(cubo);
    }
}
```

a) [4 pts.] Indique la salida que genera el método recorreArreglo para el ejemplo de prueba (cubo).

R:// 18, 17, 16, 12, 11, 10, 6, 5, 4, 15, 14, 13, 9, 8, 7, 3, 2, 1

b) [8 pts.] Haga una versión del método recorreArreglo (la misma salida) utilizando solo un ciclo.

R://

```
public static void recorreArreglo(int[][][] arreglo) {

     int nt = arreglo.length;        // cantidad de tablas
     int nf = arreglo[0].length;     // cantidad de filas
     int nc = arreglo[0][0].length;  // cantidad de columnas

     for (int k = 17;k >= 0;k--){
         System.out.println(arreglo[(k%(nt*nc))/nc][k/(nt*nc)][k%nc]);
     }
}
```

3. [10 pts.] El siguiente método determina si una hilera es correcta en cuanto a los niveles de paréntesis redondos. Haga una versión recursiva del mismo.

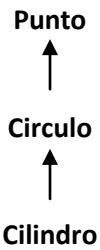
```
    public boolean parentesisCorrecto(String hilera) {
        int cont = 0;
        int i = 0;
        while (cont >= 0 && i < hilera.length()) {
            if (hilera.charAt(i) == '(') {
                cont++;
            } else {                // se asume que el caracter es ')'
                cont--;
            }
            i++;
        }
        return (cont == 0);
    }
```

R://

```
   public static boolean parentesisCorrecto(String hilera, int i, int cont) {
        boolean resultado = (cont == 0);

        if (i < hilera.length() && cont >= 0) {
            if (hilera.charAt(i) == '(') {
                resultado = parentesisCorrecto(hilera, i + 1, cont + 1);
            } else {
                resultado = parentesisCorrecto(hilera, i + 1, cont - 1);
            }
        }
        return resultado;
    }
```

4. [12 pts.] Considere la siguiente jerarquía de clases:



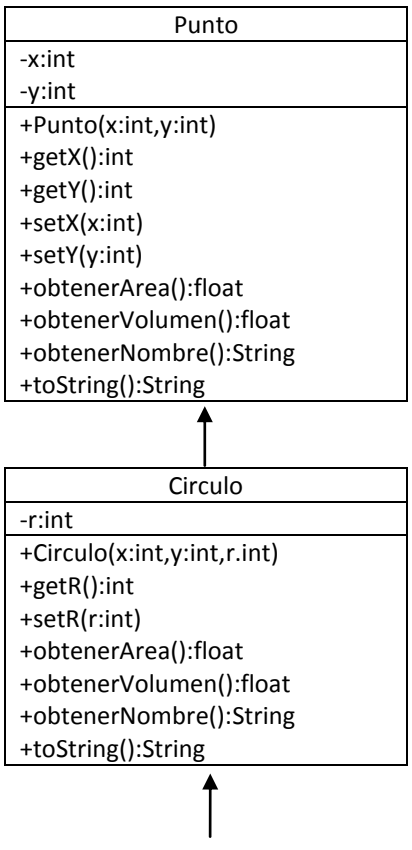
Las clases proporcionan tres métodos: obtenerArea, obtenerVolumen y obtenerNombre. La siguiente tabla muestra el resultado deseado para cada método, incluyento el toString.

	obtenerArea	**obtenerVolumen**	**obtenerNombre**	**toString**
Punto	0.0	0.0	"Punto"	[x,y]
Circulo	Πr^2	0.0	"Circulo"	centro = [x,y]; radio = r

Cilindro	$2\Pi r^2 + 2\Pi rh$	$\Pi r^2 h$	"Cilindro"	centro = [x,y];
				radio = r;
				altura = h

a) [8 pts.] Haga el diagrama de clases (UML), solo considere los atributos y métodos necesarios, (incluya setters y getters).

R://



Cilindro
-h:int
+Cilindro(x:int,y:int,r:int,h:int) +getH():int +setH(h:int) +obtenerArea():float +obtenerVolumen():float +obtenerNombre():String +toString():String

b) [4 pts.] Implemente la clase Cilindro, considere la máxima reutilización de código.

R://

```
public class Cilindro extends Circulo {

    private int h;

    public Cilindro(int x,int y,int r,int h) {
        super(x,y,r);
        this.h = h;
    }

    public int getH() {
        return h;
    }

    public void setH(int h) {
        this.h = h;
    }

    @Override
    public float obtenerArea() {
         return 2*Math.PI*(Math.pow(getR(),2)+getR()*h);
    }

    @Override
    public float obtenerVolumen() {
        return Math.PI*Math.pow(getR(),2)*h;
    }

    @Override
    public String obtenerNombre() {
         return "Cilindro";
    }

    @Override
    public String toString() {
        return super.toString() + ";\n"+"altura = " + h;
    }

}
```

Duración: 60 minutos