UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Programación Orientada a Objetos 1

Práctica Calificada 1
Pregrado
2021-I
Profesor: Rubén Rivas
Lab. 1.01

Indicaciones específicas:

- Esta evaluación contiene 6 páginas (incluyendo esta página) con 3 preguntas. El total de puntos son 20.
- El tiempo límite para la evaluación es 100 minutos.
- Cada pregunta deberá ser respondida en su correspondiente par de archivos, cabecera y fuentes, con el número de la pregunta. Por ejemplo:
  1. `p1.cpp`, `p1.h`
  2. `p2.cpp`, `p2.h`
  3. `p3.cpp`, `p2.h`
- Deberás subir estos archivos, individuales o comprimidos en un archivo **ZIP**, directamente a www.gradescope.com.

Competencias y criterios de desempeño:

- Para los alumnos de la carrera de Ciencia de la Computación

  Aplica conocimientos de computación apropiados para la solución de problemas definidos y sus requerimientos en la disciplina del programa. (nivel 2)

  Diseña, implementa y evalúa soluciones a problemas complejos de computación. (nivel 2)

  Crea, selecciona, adapta y aplica técnicas, recursos y herramientas modernas para la práctica de la computación y comprende sus limitaciones. (nivel 2)

- Para los alumnos de las carreras de Ingeniería

  Capacidad para aplicar conocimientos de matemática. (nivel 2)

  Capacidad para diseñar un sistema, un componente o un proceso para satisfacer las necesidades deseadas dentro de restricciones realistas. (nivel 2)

Calificación:

Tabla de puntos (sólo para uso del professor)

Question	Points	Score
1	7	
2	7	
3	6	
Total:	20	

1. (7 points) **Ecuación Cuadrática**
   Crear un programa que reciba 3 valores del tipo double **(a, b y c)** y que calcule las raíces de una ecuación cuadrática $ax^2 + bx + c$ y que retorne la raíz de mayor valor, en caso las raíces sean imaginarias el programa retornará **"no hay raíces reales"**.

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Listing 1: Input Format

```
1
-2
1
```

- El ingreso de los valores no requiere utilizar etiquetas (std::cout)

Listing 2: Output Format

```
1
```

Algunos ejemplos de diálogo de este programa serían:

Ejemplo 1

```
-1
4
-2
```

Output

```
3.41421
```

Ejemplo 2

```
2
4
3
```

Output

```
no hay raices reales
```

La rúbrica para esta pregunta es:

Criterio	Logrado	Parcialmente Logrado	No Logrado
Algoritmo y Código	El algoritmo y código es preciso y finito y hace exactamente lo que el enunciado requiere. (`4pts`)	Es preciso, finito y hace la mitad o más de lo que el enunciado requiere. (`2pts`)	Hace menos de la mitad de lo que el enunciado requiere (`0pts`).
Sintaxis	No existen errores sintácticos o de compilación (`2pts`)	Existen algunos errores sintácticos o de compilación. (`1pts`).	El código no compila (`0pts`).
Optimizacion	El código es óptimo y eficiente (`1pts`)	El codigo es optimizable en algunas partes (`0.5pts`).	El codigo es redundante y/o no es optimo (`0pts`).

2. (7 points) **Números triangulares**
Un número triángulo es aquel número que se obtiene de la suma de los **n** primeros números de naturales, asi por ejemplo 6 es un número triangular formado por la suma de **1 + 2 + 3** ó el número triangular **10 = 1 + 2 + 3 + 4**. Escribir un programa que solicite un **numero** y una función que permita verificar si el número ingresado es triangular o no, el programa debe de retornar si el número es triangular o no.

Listing 3: Input Format

```
630
```

- El ingreso de los valores no requiere utilizar etiquetas (std::cout)

Listing 4: Output Format

```
triangular
```

Algunos ejemplos de diálogo de este programa serían:

Ejemplo 1

```
500
```

Output

```
no triangular
```

Ejemplo 2

```
406
```

Output

```
triangular
```

La rúbrica para esta pregunta es:

Criterio	Logrado	Parcialmente Logrado	No Logrado
Código	Ha implementado funciones en forma correcta y lógica (`4pts`)	Existen algunos errores menores en la implementación (`2pts`)	El diseño y la implementacion del código no son correctos (`0pts`).
Sintaxis	No existen errores sintácticos o de compilación (`2pts`)	Existen algunos errores sintácticos o de compilación. (`1pts`).	El código no compila (`0pts`).
Optimizacion	El código es óptimo y eficiente (`1pts`)	El código es optimizable en algunas partes (`0.5pts`).	El código es redundante y/o no es óptimo (`0pts`).

3. (6 points) **Número piramidal cuadrado**
Escribir una **función recursiva** cuyo nombre sea **eliminar_vocales** que reciba como parámetro un valor del tipo **std::string** que definirá un **texto** de entrada y a partir del generar un nuevo texto que remueva todas las vocales del texto original, el nuevo texto debera ser devuelto en el valor de retorno de la función.

Listing 5: Input Format

```
Este texto tiene vocales
```

- El ingreso de los valores no requiere utilizar etiquetas (std::cout)

Listing 6: Output Format

```
st txt tn vcls
```

Algunos ejemplos de diálogo de este programa serían:

Ejemplo 1

```
Murcielago
```

Output

```
Mrclg
```

Ejemplo 2

```
Universidad de Ingenieria y Tecnologia - UTEC
```

Output

```
nvrsdd d ngnr y Tcnlg - TC
```

La rúbrica para esta pregunta es:

Criterio	Logrado	Parcialmente Logrado	No Logrado
Código	Ha implementado funciones recursivas en forma correcta y lógica (`4pts`)	Existen algunos errores menores en la implementación (`2pts`)	El diseño y la implementacion del código no son correctos (`0pts`).
Sintaxis	No existen errores sintácticos o de compilación (`1.0pts`)	Existen algunos errores sintácticos o de compilación. (`0.5pts`).	El código no compila (`0pts`).
Optimizacion	El código es óptimo y eficiente (`1pts`)	El código es optimizable en algunas partes (`0.5pts`).	El código es redundante y/o no es óptimo (`0pts`).