UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Programación II
Práctica Calificada 1
Pregrado
2022-I
Profesor José Fiestas
Lab 2.01

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 un solo archivo con el número de la pregunta.
  - `p1.cpp`
  - `p2.cpp`
  - `p3.cpp`
- Deberás subir estos archivos directamente a www.gradescope.com, uno en cada ejercicio. También puedes crear un .zip

Competencias:

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

  Aplicar conocimientos de computación y de matemáticas apropiadas para la disciplina. (Evaluar)

  Analizar problemas e identificar y definir los requerimientos computacionales apropiados para su solución.(Usar)

  Utilizar técnicas y herramientas actuales necesarias para la práctica de la computación. (Usar)

- Para los alumnos de las carreras de Ingeniería

  Capacidad de aplicar conocimientos de matemáticas (nivel 3)

  Capacidad de aplicar conocimientos de ingeniería(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	6	
2	7	
3	7	
Total:	20	

1. (6 points) **Fórmula de notas**
El curso Programacion II está implementando una nueva fórmula para calcular el promedio final del curso:

$$F = 0.15 \cdot EC + 0.2 \cdot P + 0.3 \cdot EP + 0.3 \cdot EF + 0.05 \cdot R$$

Donde EC: Evaluación Contínua, P: Proyecto, EP: Exámen Parcial, EF: Exámen Final, R: Reto, F: Nota Final
Ademas se rige por las siguientes condiciones

- si alguna de las notas es desaprobatoria (< 10.5), se desaprueba el curso con F=10
- si alguna de las notas está en el rango $[10.5 - 14]$, no se considera la nota R (Reto)
- si todas las notas son mayores que 14, se aplica la fórmula original

El código debe permitir ingresar todas las notas e imprimir el promedio final F en una tabla (el formato lo decide el alumno). El promedio debe redondearse al entero superior si la fracción es ≥ 0.5

La rúbrica para esta pregunta es:

Criterio	Excelente	Adecuado	Mínimo	Insuficiente
Ejecución	El diseño del algoritmo es ordenado y claro, siguiendo buenas prácticas en programación. La ejecución es correcta (`3pts`)	El diseño del algoritmo es ordenado y claro. La ejecución es correcta (`2pts`)	La ejecución es correcta (`1pts`).	La ejecución no es correcta (`0.5pts`)
Sintaxis	No existen errores sintácticos o de compilación (`2pts`)	Existen algunos errores sintácticos de menor relevancia, que no afectan el resultado (`1.5pts`).	Existen errores sintácticos en la forma de ejecución, que no afectan el resultado (`1pts`).	El código tiene errores de sintáxis que afectan el resultado (`0.5pts`).
Optimizacion	El código es óptimo y eficiente. De buen performance e interacción con el usuario (`2pts`)	El codigo es de buen performance durante la ejecución (`1.5pts`).	El codigo no está optimizado pero la ejecución no es deficiente(`1pts`).	El codigo no está optimizado y la ejecución es deficiente (`0pts`).

2. (7 points) **Crackeo de clave**
Escriba un código en C++ que genere una clave de 8 dígitos binarios en forma aleatoria. Ahora escriba una función que genere una clave también en forma aleatoria y compare ésta con la clave original.
Ya que muy probablemente no la encuentre al primer intento, calcule e imprima la cantidad de intentos que necesita para encontrar la clave.
Una vez encontrada la clave, traduzca la secuencia de unos y ceros de acuerdo a las siguientes reglas:
- el primer 1 corresponde a la letra u, el segundo 1 a la letra o , el tercer uno a la i, el cuarto a la e, y el quinto a la a. Si aparecen más unos, se regresa a la u.
- el primer 0 corresponde a la letra z, el segundo 0 a la letra y, el tercer 0 a la x, el cuarto a la w, y el quinto a la v. Si aparecen más 0s, se regresa a la z.
Imprima la clave codificada (traducida)
Ejemplos:

```
Clave encontrada despues de 313 intentos: 10101111
Clave traducida: uzoyieau
```

```
Clave encontrada despues de 530 intentos: 00111001
Clave traducida: zyuoixwe
```

La rúbrica para esta pregunta es:

Criterio	Excelente	Adecuado	Mínimo	Insuficiente
Ejecución	El diseño del algoritmo es ordenado y claro, siguiendo buenas prácticas en programación. La ejecución es correcta (`2pts`)	El diseño del algoritmo es ordenado y claro. La ejecución es correcta (`1.5pts`)	La ejecución es correcta (`1pts`).	La ejecución no es correcta (`0.5pts`)
Sintaxis	No existen errores sintácticos o de compilación (`2pts`)	Existen algunos errores sintácticos de menor relevancia, que no afectan el resultado (`1.5pts`).	Existen errores sintácticos en la forma de ejecución, que no afectan el resultado (`1pts`).	El código tiene errores de sintáxis que afectan el resultado (`0.5pts`).
Optimizacion	El código es óptimo y eficiente. De buen performance e interacción con el usuario (`2pts`)	El codigo es de buen performance durante la ejecución (`1.5pts`).	El codigo no está optimizado pero la ejecución no es deficiente(`1pts`).	El codigo no está optimizado y la ejecución es deficiente (`0pts`).

3. (7 points) **Modelo de LEGO**
Un nuevo modelo de LEGO es una torre con piezas de color rojo (**r**), azul (**a**) y blanco (**b**). La secuencia correcta de colores en la torre esta encargada a dos robots (A y B). El robot A recibe siempre la primera pieza, y ambos se rigen por las siguientes reglas.
- Si el robot A recibe una pieza

   - Si la pieza es azul, el robot A mismo ensambla la siguiente pieza
   - Si la pieza es roja, el robot B ensambla la siguiente pieza
   - Si la pieza es de otro color (o no valida), se detiene el ensamblaje con mensaje de error

   - Si el robot B recibe una pieza

   - Si la pieza es blanca, el robot B mismo ensambla la siguiente pieza
   - Si ya no hay mas piezas, el robot B empaca el modelo y lo envía a almacén
   - Si la pieza es de otro color (o no valida), se detiene el ensamblaje con mensaje de error

   Escriba un codigo en C++ utilizando funciones con punteros, que ejecute el ensamblado de las piezas, en el cual el input es una secuencia de caracteres que representan los colores de las piezas. Las torres deben tener 3 piezas.

   Ejemplos:

   Ingrese la secuencia de piezas de colores:
   arb
   Torre ensamblada y enviada a almacén

   Ingrese la secuencia de piezas de colores:
   bar
   Torre rechazada. No puedo construir una torre con esa secuencia de colores según mi programa de trabajo. Firmado: robots A y B :)

   La rúbrica para esta pregunta es:

Criterio	Excelente	Adecuado	Mínimo	Insuficiente
Ejecución	El diseño del algoritmo es ordenado y claro, siguiendo buenas prácticas en programación. La ejecución es correcta (`3pts`)	El diseño del algoritmo es ordenado y claro. La ejecución es correcta (`2pts`)	La ejecución es correcta (`1pts`).	La ejecución no es correcta (`0.5pts`)
Sintaxis	No existen errores sintácticos o de compilación (`2pts`)	Existen algunos errores sintácticos de menor relevancia, que no afectan el resultado (`1.5pts`).	Existen errores sintácticos en la forma de ejecución, que no afectan el resultado (`1pts`).	El código tiene errores de sintáxis que afectan el resultado (`0.5pts`).
Optimizacion	El código es óptimo y eficiente. De buen performance e interacción con el usuario (`2pts`)	El codigo es de buen performance durante la ejecución (`1.5pts`).	El codigo no está optimizado pero la ejecución no es deficiente(`1pts`).	El codigo no está optimizado y la ejecución es deficiente (`0pts`).