UNIVERSIDAD
DE GRANADA

Doble Grado en Informática y Matemáticas

Fundamentos de Programación

Examen práctico 2. Curso 2019/2020

A TENER EN CUENTA

- Cada alumno debe encender un ordenador del aula e insertar su usuario y password junto con el código `examenFPprado`. Esta imagen no tiene acceso a internet (salvo a PRADO y a `decsai.ugr.es`) ni a los puertos USB. Si no pudiera entrar en su cuenta puede utilizar la genérica con el usuario `generica` y password `temporal`.
- Los ficheros han de entregarse a través de la plataforma PRADO. Si no fuera posible acceder a PRADO, deben entregarse en la plataforma docente en `decsai.ugr.es`, o alternativamente en `https://150.214.191.180`.
- Los ficheros a entregar deben llamarse, obligatoriamente, `subvector.cpp`, `circunferencia.cpp`, `matriz.cpp`, `rotarbits.cpp`. No se deben comprimir en un archivo `.zip`. Sólo deben entregarse ficheros `.cpp`.
- En la cabecera de cada fichero se deben incluir los siguientes datos, como un comentario del programa:

```
// Nombre: <Nombre y Apellidos del alumno>
// Problema: <subvector o circunferencia o matriz o rotarbits>
```

- En la evaluación de los ejercicios se tendrá en cuenta, además de la corrección de la solución propuesta, el estilo de programación, el uso correcto de espacios y tabuladores, así como la claridad del código fuente y de los comentarios. Es decir, se tendrán en cuenta las buenas prácticas de programación explicadas en las transparencias de teoría.
- Para resolver los ejercicios sólo pueden usarse las herramientas explicadas en las clases de teoría y prácticas.
- Los ejercicios se pueden entregar tantas veces como se quiera durante el examen. De hecho, se recomienda que se entreguen varias veces a lo largo del examen ya que, si el ordenador se quedara "colgado", habría que reiniciarlo y se perdería toda la información. Lo ideal es tener el explorador abierto en la página de entrega de PRADO para, cuando se necesite, "soltar" rápidamente el archivo en la zona de subida y actualizar la entrega.
- Este examen se corresponde con un 50 % (2 puntos de 4) de la nota de prácticas. Se puede entregar cualquier combinación de los cuatro ejercicios (en particular, los cuatro, ya que PRADO permite subir hasta cuatro archivos). La máxima nota se obtiene si el examen se califica con 10 o más puntos. Los puntos que se obtengan a partir de 10 servirán para compensar posibles fallos en el examen teórico.
- Tiempo de la prueba: 1 hora y 30 minutos (de 15:45 horas a 17:15 horas)

---

1. **Subvector (3 puntos).** Implementar un programa que, dado un vector de enteros, muestre por pantalla el subvector cuya suma de sus elementos sea máxima. Por ejemplo, si el vector es `{1, 2, -5, 4, 1,-3, 2}`, entonces el subvector es `{4,1}`, que suma 5. Por subvector se entiende un conjunto de componentes consecutivas del vector original.

2. **Circunferencia (3 puntos)** Implementar una clase `Circunferencia` para representar una circunferencia en el plano. En particular,

   - diseñar la clase utilizando unos datos miembro adecuados (1 punto).
   - implementar un constructor por defecto que inicialice a la circunferencia unidad, y un constructor con parámetros (1 punto)
   - implementar un método que decida si dos circunferencias se intersecan o no (1 punto).

   Implementar una función `main` para probar la clase. La función `main` no se evaluará.

3. **Matriz (4 puntos)** En una matriz de caracteres de dimensión $f \times c$ puede haber, en cada casilla, el caracter "X", o bien el caracter ".". Diseñar y escribir un programa que encuentre la mayor secuencia consecutiva, horizontal o vertical, de puntos existente en la matriz. Para ello, debe devolver la fila y columna donde comienza la secuencia y su orientación: "h" para orientación horizontal y "v" para vertical. En caso de empate entre una secuencia horizontal y otra vertical, se devolverá preferentemente la secuencia con orientación horizontal. En caso de empate entre dos secuencias horizontales, se devolverá la de menor valor de fila. En caso de empate entre dos secuencias verticales, se devolverá la de menor valor de columna. En caso de que la matriz no contenga ningún punto, debe devolver `fila=-1`, `col=-1` y `orientacion=h`..

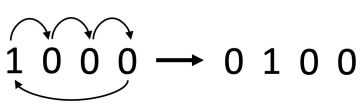
   Por ejemplo, para la siguiente matriz 6×6, el programa debe encontrar `fila=1`, `col=0` y `orientacion=v` (donde existe una secuencia de puntos de longitud 4).

X	.	X	X	.	X
.	X	X	X	X	X
.	.	X	.	X	.
.	X	X	X	X	X
.	X	.	.	X	.
X	X	X	X	X	X

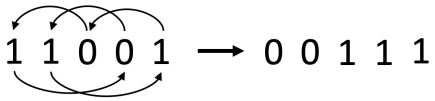
   Se recomienda crear un fichero `.txt`, con las dimensiones de la matriz y la propia matriz, y redireccionar la entrada para realizar las pruebas del programa más cómodamente.

4. **RotarBits (5 puntos).** Implementar una función
   `int RotarBits(int numero, int veces, bool direccion);`
   que devuelva el número que se obtiene al desplazar los bits/dígitos de la representación en binario del entero positivo `numero` tantas posiciones como indica `veces` en la dirección que indica `direccion` (`true` para la derecha, `false` para la izquierda), donde la traslación es cíclica (el siguiente al último es el primero). Por ejemplo,

   - `RotarBits(8,1,true)` devuelve 4, ya que 8≡1000, y al desplazar los bits una posición a la derecha, se obtiene 0100, que es 4.



   - `RotarBits(25,2,false)` devuelve 7, ya que 25≡11001, y al desplazar los bits dos posiciones a la izquierda, se obtiene 00111, que es 7.



   Implementar una función `main` para probar la función. La función `main` no se evaluará.