UNIVERSITAT JAUME·I

1º Ingeniería Informática
1º Matemática Computacional

EI1008/MT1008 Programación II

Práctica 1: Revisión de conceptos básicos de programación

Ejercicio 1

Escribe un programa que lea un número entero, n, y escriba su factorial, $n!$. Un ejemplo de ejecución del programa es:

```
Introduce un número entero: 5
5! = 120
```

El factorial de 15 es 1307674368000. ¿Es ese el resultado que obtienes al ejecutar tu programa? Si no lo es, piensa a qué puede deberse el error producido.

Ejercicio 2

Escribe un programa que lea un número entero, n, y escriba su doble factorial, $n!!$. El doble factorial de n se define como:

$$n!! = \begin{cases} 1, & \text{si } n = 0 \text{ o } n = 1 \\ 2 \times 4 \times 6 \times \ldots \times (n-2) \times n, & \text{si } n \text{ es par} \\ 1 \times 3 \times 5 \times \ldots \times (n-2) \times n, & \text{si } n \text{ es impar} \end{cases}$$

Un ejemplo de ejecución del programa es:

```
Introduce un número entero: 10
10!! = 3840
```

Otro ejemplo de ejecución del programa es:

```
Introduce un número entero: 7
7!! = 105
```

Ejercicio 3

Escribe un programa que lea un número entero y escriba "Es primo" o "No es primo", según corresponda. Un ejemplo de ejecución del programa es:

```
Introduce un número entero: 167
Es primo
```

Otro ejemplo de ejecución del programa es:

```
Introduce un número entero: 1003
No es primo
```

Ejercicio 4

Escribe un programa que lea un número entero, n, y escriba todos los números primos menores que n. Un ejemplo de ejecución del programa es:

```
Introduce un número entero: 23
Los números primos menores que 23 son: 2 3 5 7 11 13 17 19
```

Ejercicio 5

Escribe un programa que lea un número entero, n, y escriba su primorial, $n\#$. El primorial de n se define como el producto de todos los números primos menores o iguales a n. Un ejemplo de ejecución del programa es:

```
Introduce un número entero: 11
11# = 2310
```

Otro ejemplo de ejecución del programa es:

```
Introduce un número entero: 15
15# = 30030
```

Ejercicio 6

Escribe un programa que lea un número entero, n, y escriba el mayor número comprendido entre 1 y n con más divisores. Un ejemplo de ejecución del programa es:

```
Introduce un número entero: 100
El número con más divisores es 96 (12 divisores)
```

Ejercicio 7

Escribe un método estático, `esPrimo`, que reciba como parámetro un número entero y devuelva como resultado `true` cuando ese número sea primo y `false` cuando no lo sea.

Reescribe el programa del ejercicio 3 para que haga uso del método `esPrimo`.

Ejercicio 8

Reescribe el programa del ejercicio 4 para que haga uso del método `esPrimo`.

Ejercicio 9

Reescribe el programa del ejercicio 5 para que haga uso del método `esPrimo`.

Ejercicio 10

Escribe un método estático, `contarDivisores`, que reciba como parámetro un número entero y devuelva como resultado la cantidad de divisores de ese número.

Reescribe el programa del ejercicio 6 para que haga uso del método `contarDivisores`.

En los siguientes ejercicios se pide la implementación de diferentes métodos estáticos. En todos ellos deberás escribir también un método `main` que haga uso de los métodos implementados para probar su correcto funcionamiento mediante una pequeña batería de pruebas.

Ejercicio 11

Escribe un método estático, `esSufijo`, que reciba como parámetros dos cadenas y devuelva `true` cuando la segunda cadena sea un sufijo de la primera y `false` cuando no lo sea. Los únicos métodos de cadenas que puedes usar son `length` y `charAt`.

Ejercicio 12

Escribe un método estático, `contarPalabras`, que reciba como parámetro una cadena y devuelva la cantidad de palabras que aparecen en la misma. Consideraremos que una palabra está formada por una secuencia de caracteres distintos del espacio en blanco. Ten en cuenta que pueden existir varios espacios entre dos palabras sucesivas, así como al comienzo y al final de la cadena. Los únicos métodos de cadenas que puedes usar son `length` y `charAt`.

Ejercicio 13

Escribe un método estático, `obtenerPalabra`, que reciba como parámetros una cadena y un número entero `i` y devuelva la `i`-ésima palabra de la cadena cuando el valor de `i` sea válido (es decir, esté comprendido entre 1 y el número de palabras de la cadena) y `null` cuando no lo sea. Los únicos métodos de cadenas que puedes usar son `length`, `charAt` y `substring`.

Ejercicio 14

Escribe un método estático, `últimaPosición`, que reciba como parámetros un vector de enteros y un número entero y devuelva la última posición del vector que contenga el elemento dado o un valor negativo cuando el elemento no aparezca en el vector.

Ejercicio 15

Escribe un método estático, `estáOrdenado`, que reciba como parámetro un vector de cadenas y devuelva `true` cuando el vector esté ordenado lexicográficamente de menor a mayor y `false` cuando no lo esté.

Ejercicio 16

Escribe un método estático, `contarOlasDeFrío`, que reciba como parámetros un vector de números de tipo `double` y un número entero `n`. Los valores del vector dado representan las temperaturas máximas de cada día en una ciudad a lo largo de un año. El método debe devolver la cantidad de olas de frío que sufrió la ciudad en ese año. Se considera que hay una ola de frío cuando la temperatura máxima está por debajo de 0 grados durante más de `n` días consecutivos.

Ejercicio 17

Escribe un método estático, `hayRepetidos`, que reciba como parámetro un vector de enteros y devuelva `true` cuando en el vector haya elementos repetidos y `false` en caso contrario.

Ejercicio 18

Escribe un método estático, `eliminarPosición`, que reciba como parámetros un vector de enteros y un entero que representa una posición en el vector y devuelva un nuevo vector con los mismos elementos que el vector dado, pero en el que se haya eliminado el elemento que ocupaba la posición indicada. Si la posición indicada no fuera válida, el método debe devolver el mismo vector que ha recibido como parámetro.

Ejercicio 19

Escribe un método estático, `eliminarValor`, que reciba como parámetros un vector de enteros y un número entero y devuelva un nuevo vector que contenga los mismos elementos que el vector recibido, salvo aquellos que coincidan con el número dado.

Ejercicio 20

Escribe un método estático, `contiene`, que reciba como parámetros un vector de enteros y un número entero y devuelva `true` cuando el número aparezca en el vector y `false` en caso contrario.

Haciendo uso de ese método, escribe otro método estático, `contiene`, que reciba como parámetros dos vectores de enteros y devuelva como resultado `true` cuando todos los elementos del segundo vector aparezcan en el primero y `false` en caso contrario. Puedes suponer que en los vectores no hay elementos repetidos.

Ejercicio 21

Escribe un método estático, `posiciónInserción`, que reciba como parámetros un vector de enteros ordenado de menor a mayor y un valor entero y devuelva la posición en la que debería insertarse el valor dado para que el vector siguiese estando ordenado. Si el valor dado ya estuviera en el vector, se debería devolver la primera posición en la que se encontrase.

Ejercicio 22

Escribe un método estático, `másOlasDeFrío`, que reciba como parámetros una matriz de números de tipo `double` y un entero `n`. Al igual que en el ejercicio 16, los valores de cada fila de la matriz representan las temperaturas máximas de cada día en una ciudad a lo largo de un año y se considera que hay una ola de frío cuando la temperatura máxima es negativa durante más de `n` días consecutivos. En este caso, cada fila de la matriz almacena las temperaturas de diferentes años, comenzando en 1900. Por simplicidad, puedes considerar que todos los años tienen el mismo número de días. El método debe devolver como resultado el año más reciente en el que se produjo un mayor número de olas de frío. Si no se hubiese producido ninguna ola de frío, se debería devolver un número negativo.

Para solucionar este ejercicio, debes emplear el método implementado en el ejercicio 16.

Ejercicio 23

Escribe un método estático, `estáEnTodasLasColumnas`, que reciba como parámetros una matriz de números enteros y un número entero y devuelva `true` cuando el número dado aparezca al menos una vez en todas y cada una de las columnas de la matriz y `false` en caso contrario.

Ejercicio 24

El vector `equipos` contiene los nombres de los equipos que participan en la liga de fútbol de primera división femenina durante la temporada 2012/2013. Los resultados de todos los partidos disputados hasta la jornada actual se almacenan en el fichero de texto `liga.txt`. Cada línea del fichero representa el resultado de un partido y contiene los siguientes datos: el nombre del equipo local, el número de goles que ha marcado, el nombre del equipo visitante y el número de goles que ha marcado. Cada dato se separa del siguiente por un espacio en blanco. Los nombres de los equipos no contienen espacios en blanco. Un ejemplo del contenido del fichero sería:

```
Athletic_Club 5 SD_Lagunak 0
SC_Huelva 1 Athletic_Club 1
SD_Lagunak 0 SC_Huelva 5
```

Escribe un método estático, `crearMatrizResultados`, que reciba como parámetros el vector de equipos y el nombre del fichero de resultados. El método debe devolver una matriz de $N \times N$ caracteres (siendo N el número de equipos), donde cada celda `[i][j]` represente el resultado del partido disputado entre el equipo local `equipos[i]` y el equipo visitante `equipos[j]`. La celda debe almacenar:

- `'1'`, si el partido lo ha ganado el equipo local,
- `'X'`, si el partido ha terminado en empate,
- `'2'`, si el partido lo ha ganado el equipo visitante,
- `'-'`, si el partido no se ha jugado.

Por ejemplo, para el siguiente vector de equipos:

"Athletic_Club"	"SC_Huelva"	"SD_Lagunak"
0	1	2

y el fichero de ejemplo, la función debería devolver una matriz como la siguiente:

	0	1	2
0	'-'	'-'	'1'
1	'X'	'-'	'-'
2	'-'	'2'	'-'

Ejercicio 25

Escribe un método estático, `másVictoriasFuera`, que reciba como parámetros una matriz de resultados como la obtenida en el ejercicio 24 y el vector de equipos y devuelva el nombre del equipo que ha obtenido más victorias jugando como equipo visitante. En caso de empate entre varios equipos, puedes devolver cualquiera de ellos.

Ejercicio 26

Escribe un método estático, `obtenerPuntos`, que reciba como parámetros una matriz de resultados como la obtenida en el ejercicio 24, el vector de equipos y el nombre de un equipo y devuelva la cantidad de puntos que ha obtenido el equipo dado. Esta cantidad se calcula sumando tres puntos por cada partido ganado y un punto por cada partido empatado.