Ejercitación № 4: Ciclos

Cátedra Programación II Septiembre 2016

1. Bucles While

Desarrolle cada uno de los siguientes ejercicios detallando cada una de las etapas involucradas en la construcción de un programa. Para la etapa de prueba de nuestro programa se deberá emplear la aplicación `pytest`. Tener en cuenta que deberá crear funciones que permitan el pasaje de argumentos para posteriormente testear el programa contra los datos de testing. Cuando sea posible, proponga resolver los problemas con funciones recursivas, como venimos haciendo en teoría, y compare ambas soluciones.

Ejercicio 1. Nos piden que escribamos una función que le pida al usuario que ingrese un número positivo. Si el usuario ingresa cualquier cosa que no sea lo pedido se le debe informar de su error mediante un mensaje y volverle a pedir el número.

Ejercicio 2. Escriba un programa que permita al usuario ingresar un conjunto de notas, preguntando a cada paso si desea ingresar más notas, e imprimiendo el promedio correspondiente, al finalizar la toma de datos.

Ejercicio 3. Escriba un programa que pida dos números enteros. El programa pedirá de nuevo el segundo número mientras no sea mayor que el primero. El programa terminará escribiendo los dos números.

Ejercicio 4. Escriba una función que reciba dos números como parámetros, y devuelva cuántos múltiplos del primero hay, que sean menores que el segundo.

a) Implementarla utilizando un ciclo `for`, desde el primer número hasta el segundo.

b) Implementarla utilizando un ciclo `while`, que multiplique el primer número hasta que sea mayor que el segundo.

c) Comparar ambas implementaciones: ¿Cuál es más clara? ¿Cuál realiza menos operaciones?

Ejercicio 5. Manejo de contraseñas

a) Escribir un programa que contenga una contraseña inventada, que le pregunte al usuario la contraseña, y no le permita continuar hasta que la haya ingresado correctamente.

b) Modificar el programa anterior para que solamente permita una cantidad fija de intentos.

c) Modificar el programa anterior para que sea una función que devuelva si el usuario ingresó o no la contraseña correctamente, mediante un valor booleano (`True` o `False`).

Ejercicio 6. Escriba una función que reciba un número natural e imprima todos los números primos que hay hasta ese número. Para esto se pide que:

a) Defina una función `es_primo` que toma un número natural y verifique si es un número primo.

b) Resuelva el problema usando la función definida en el punto anterior.

Ejercicio 7. **Potencias de dos**

a) Escribir una función `es_potencia_de_dos` que reciba como parámetro un número natural, y devuelva `True` si el número es una potencia de 2, y `False` en caso contrario.

b) Escribir una función que, dados dos números naturales pasados como parámetros, devuelva la suma de todas las potencias de 2 que hay en el rango formado por esos números (0 si no hay ninguna potencia de 2 entre los dos). Utilizar la función `es_potencia_de_dos`, descripta en el punto anterior.

2. Bucles For con Listas y Strings

Resuelva los siguientes ejercicios aplicando lo visto en teoría para esto

Ejercicio 8. Vamos a escribir un programa que permita crear una lista de palabras, para esto el programa tiene que:

a) pedir un número `numeroPalabras` que va a representar la cantidad de palabras que va a tener la lista

b) solicitar que se ingresen las `numeroPalabras` que se ingresó en el punto anterior. Para esto, recuerde que vimos el operador + para agregar elementos a una lista.

c) luego de que se ingresen todas las palabras se debe mostrar la lista de palabras obtenida

Ejercicio 9. Defina una función `buscar` que toma una lista de palabras `listaPalabras` como parámetro y:

a) solicita que se ingrese una palabra `palabraABuscar`

b) se verifica si `palabraABuscar` ingresada se encuentra presente en `listaPalabras`

c) se muestra un cartel indicando si está o no la palabra buscada

Ejercicio 10. Defina una función `cuentaApariciones` que toma una lista de palabras `listaPalabras` como parámetro y:

a) solicita que se ingrese una palabra `palabraABuscar`

b) se cuenta la cantidad de veces que `palabraABuscar` esté presente en `listaPalabras`

c) se muestra un cartel indicando la cantidad de veces que esta presente o, si no se encuenttra

Ejercicio 11. Construya un programa el cual sobre una lista de números ingresados por el usuario, y un número n también dado, calcule la cantidad de elementos de la lista ingresada que son mayores a n. Utilizar funciones por separado para el ingreso de los datos en el arreglo y el cómputo de elementos mayor que el dado dado.

Ejercicio 12. Defina una función `cuentaAparicionesCadena` que toma un string `palabra` como parámetro y:

a) solicita que se ingrese un caracter `caracterABuscar`

b) se cuenta la cantidad de veces que `caracterABuscar` esté presente en `palabra`

c) se muestra un cartel indicando la cantidad de veces que esta presente o, si no se encuenttra

d) Compare este problema con el anterior. Analice qué tuvo que cambiar entre uno y otro.

Ejercicio 13. Se pide construir un programa, el cual dada una lista que representa montos de dinero ingresados por el usuario, devuelva como resultado la suma de los montos presentes en la lista. Para esto último se pide definir una función `suma_dinero` que toma como entrada la lista con los montos de dinero y calcula dicha suma.

Ejercicio 14. Modifique el programa anterior de forma tal que:

a) Permite ingresar los valores sin corroborar en el momento si corresponden a montos de dinero o no (si son o no positivos)

b) Cree una función `montos_positivos` la cual devuelve `True` si todos los valores de la lista son positivos, y `False` en caso contrario.

c) Cree una función `checked_suma` la cual recibe como entrada una lista de números, y devuelve la suma de todos sus elementos si la lista se corresponde con una lista de montos de dinero, sino deberá devolver un error.