EJERCICIO 1:

**Imprimir todos los dígitos decimales, del 0 al 9, utilizando una repetición.**

Se utiliza una sentencia FOR con un rango con límite superior 10 (el 10 está excluido) “for i in range (10)”. Dentro de ella por cada iteración se imprime por pantalla mediante la función print() el valor de la variable “i”.

Referencia chatGpt: Sintaxis end = “ “ , para imprimir sin salto de línea luego de cada iteración.

RESULTADO OBTENIDO:

“0 - 1 - 2 - 3 - 4 - 5 - 6 - 7 - 8 - 9 –“

EJERCICIO 2:

**Imprimir todos los números entre el 100 y el 199.**

Se utiliza una sentencia FOR con un rango que inicia en el número 100 y con el límite superior en 200 (el 200 está excluido) “for i in range (100,200)”. Dentro de ella por cada iteración se imprime por pantalla mediante la función print() el valor de la variable “i”.

RESULTADO OBTENIDO:

“100 - 101 - 102 - 103 - 104 - 105 - 106 - 107 - 108 - 109 - 110 - 111 - 112 - 113 - 114 - 115 - 116 - 117 - 118 - 119 - 120 - 121 - 122 - 123 - 124 - 125 - 126 - 127 - 128 - 129 - 130 - 131 - 132 - 133 - 134 - 135 - 136 - 137 - 138 - 139 - 140 - 141 - 142 - 143 - 144 - 145 - 146 - 147 - 148 - 149 - 150 - 151 - 152 - 153 - 154 - 155 - 156 - 157 - 158 - 159 - 160 - 161 - 162 - 163 - 164 - 165 - 166 - 167 - 168 - 169 - 170 - 171 - 172 - 173 - 174 - 175 - 176 - 177 - 178 - 179 - 180 - 181 - 182 - 183 - 184 - 185 - 186 - 187 - 188 - 189 - 190 - 191 - 192 - 193 - 194 - 195 - 196 - 197 - 198 - 199 –“

EJERCICIO 3:

**Imprimir los números entre el 5 y el 20, saltando de tres en tres.**

Se utiliza una sentencia FOR con un rango que inicia en el número 5, con el límite superior en 21 (el 21 está excluido) Y con una incrementación de 3 veces por iteración, es decir, por cada vuelta la variable de control i aumenta de 3 en 3 “for in range (5,21,3)”. Dentro de ella por cada 3 iteraciones se imprime por pantalla el valor de la variable “i”.

RESULTADO OBTENIDO:

“5 - 8 - 11 - 14 - 17 - 20 –“

EJERCICIO 4:

**Requerir al usuario que ingrese un número entero positivo e imprimir todos los números correlativos entre el ingresado por el usuario y uno menos del doble del mismo.**

Defino una función que recibe un entero como parámetro, para validar que el numero ingresado sea positivo “validarNumeroPositivo(numero)”. Luego se le pide al usuario mediante la función “input()” que ingrese un número y a su vez se castea con la función int(), para que lo que ingresa el usuario sea un dato de tipo entero. Se utiliza la función para la validación de números positivos. Una vez validado, se incrementa al doble su valor asignándolo a la variable limite. Se utiliza una sentencia FOR con un rango que inicia en el número ingresado por el usuario y como límite superior la variable limite “for i in range (numero,limite)”. Dentro de ella por cada iteración se imprime por pantalla mediante la función print() el valor de la variable “i”.

TESTING:

TIPO DE DATO: Entero positivo

DATO: 8

RESULTADO OBTENIDO: 8-9-10-11-12-13-14-15

TIPO DE DATO: Entero negativo

DATO: -5

RESULTADO OBTENIDO: “no es un numero positivo”, “ingrese un numero entero positivo”

EJERCICIO 5

**Escribir un programa que solicite al usuario una cantidad y luego itere la cantidad de veces dada. En cada iteración, solicitar al usuario que ingrese un número. Al finalizar, mostrar la suma de todos los números ingresados.**

Definición de un acumulador y asignación con valor 0, “suma = 0”

Se Hace uso de la función para validar números positivos del ejercicio N°4 ya que, no se puede iterar con números negativos.

Se pide al usuario mediante la función input() el ingreso de la cantidad de veces que va a iterar el programa y se lo castea como un entero. Ese valor cantidad se lo utiliza como rango en la sentencia for, “ for in range (cantidad) “, donde se pide al usuario que ingrese un numero por cada vuelta. A su vez, se utiliza el acumulador para ir sumando los números ingresados por el usuario “suma += numero”. Finalmente se imprime por pantalla con la función print() el resultado de la sumatoria de números ingresados.

TESTING:

TIPO DE DATO: ENTERO

DATO: 3

RESULTADO OBTENIDO:

cantidad = 3

“ingrese un número:”

“ingrese un número:”

“ingrese un número:”

TIPO DE DATOS: ENTERO

DATOS: 1,2,3

RESULTADO OBTENIDO:

suma = 6

“La suma de los números ingresados es: 6”

TIPO DE DATO: ENTERO

DATO: -3

RESULTADO OBTENIDO:

“No es un numero positivo”

“Ingrese un numero entero positivo: “

EJERCICIO 6:

**Solicitar al usuario que ingrese una frase y luego imprimir un listado de las vocales que aparecen en esa frase (sin repetirlas).**

Se define una tupla que contiene las 5 vocales. Se define una función “listaDeVocales(string,tupla)” que recibe como parámetros, un string con la frase y una tupla con las vocales. Se pide al usuario mediante la función input() que ingrese una frase. Se llama a la función listaDevocales donde, mediante el uso de for anidados, se compara cada letra de la frase con cada una de las vocales que contiene la tupla. Si la letra es igual a alguna vocal entonces, se guarda en una lista “vocalesFrase”, con el método “append”, para luego utilizar un condicional que, verifica si se repite esa vocal en la lista “vocalesFrase”. En el caso que se repita dicha vocal, se elimina utilizando el método “remove”. Finalmente, con el método sort, se ordena alfabéticamente la lista “vocalesFrase”.

TESTING:

TIPO DE DATOS: STRING

DATO: “Hola que tal”

RESULTADO OBTENIDO:

“Frase: hola que tal”

“Las vocales que contiene en la frase son:

['a', 'e', 'o', 'u']”

TIPO DE DATOS: STRING

DATO: “hhh”

RESULTADO OBTENIDO:

“Frase: hhh”

“Las vocales que contiene en la frase son:

[ ]”

EJERCICIO 7:

**Solicitar al usuario que ingrese una frase y luego imprimir la cantidad de vocales que se encuentran en dicha frase.**

Funcionamiento similar al ejercicio N°6, definición de tupla de vocales, ingreso de frase por input() pero, la diferencia está en que la función que hace el reconocimiento de vocales en la frase, ahora utiliza un contador “contadorVocales”, por el cual a través del for anidado, cuenta cuantas veces aparece una vocal en esa frase. Aumentando en una unidad el contador “contadorVocales”. Luego imprime los datos por pantalla con la función print().

TESTING:

TIIPO DE DATO: string

DATO: “Que tal si salimos”

RESULTADO OBTENIDO:

“Frase: Que tal si salimos”

“La cantidad de vocales que aparecen en la frase son: 5”

TIIPO DE DATO: string

DATO: “Hhhh”

RESULTADO OBTENIDO:

“Frase: Hhhh”

“La cantidad de vocales que aparecen en la frase son: 0”

EJERCICIO 8:

**Escribir un programa que muestre la sumatoria de todos los números entre el 0 y el 100.**

Se define un acumulador “suma” y se inicializa en 0. Mediante un for, con un rango del 0 al 100 (límite superior excluido 101) “for i in range (101)”, se realiza la suma de cada iteración de la variable de control “i”, guardando el resultado en el acumulador “suma”. Luego se imprime el resultado por pantalla con la función print().

RESULTADO OBTENIDO:

“La suma de todos los números entre el 0 y el 100 es: 5050 “

EJERCICIO 9:

**Escribir un programa que muestre la sumatoria de todos los múltiplos de 3 encontrados entre el 0 y el 100.**

Funciona similar al ejercicio N°8, con la diferencia que en el for se utiliza un condicional “if i %3 == 0:” para detectar solo los números que son múltiplos de 3. Esto lo hace gracias a que se usa el operador modulo de 3 igual a 0, esto quiere decir que, si el resultado de la división entre el valor de la variable de control “i” y 3 no genera un resto, es decir sin decimales, lo guarda en el acumulador “suma”, donde se va a sumar al valor que tenia anteriormente. Luego se imprime por pantalla el resultado con la función print().

RESULTADO OBTENIDO:

“La suma de todos los múltiplos entre el 0 y el 100 es: 1683”

Nota: “Los números múltiplos de 3 que existen entre 0 y 100 son: 3,6,9,12,15,18,21,24,27,30,33,36,39,42,45,48,51,54,57,60,63,66,69,72,75,78,81,84,87,90,93,96,99. Y la suma de ellos da como resultado 1683”

EJERCICIO 10:

**Dado un número entero positivo, mostrar su factorial. El factorial de un número se obtiene multiplicando todos los números enteros positivos que hay entre el 1 y ese número.**

El factorial de un numero es el producto de todos los números enteros positivos desde 1 hasta n. Como solo se puede hacer con números enteros positivos, se vuelve a utilizar la función del ejercicio N°4 para validar números positivos “validarNumeroPositivo°.

Se inicializa la variable “factorial” en 1, si se inicializa en 0 todos los resultados posibles darian 0. Se pide ingresar un numero entero positivo con la función input() y se castea a entero con int(), se guarda el valor en la variable “numero”. Se llama a la función “validarNumeroPositivo” para verificar que el numero sea positivo. Luego, se ejecuta la sentencia for con un rango con el limite superior excluido igual a la variable numero “for i in range (numero)”. Dentro del for se ejecuta la siguiente operación:

“factorial \*= (numero-i)”, obteniendo así el factorial del numero ingresado por el usuario. Ejemplo: Factorial del numero 3, en la primera iteración la variable factorial es igual a 1, la variable de control i es igual a 0 y la variable numero es igual a 3. Por lo tanto: “factorial = 1 x (3-0)”, lo que es igual a “factorial = 3”, en las otras dos iteraciones se obtiene lo siguiente: “factorial = 3 x (3-1)”, “factorial = 6”, “factorial = 6 x (3-2)”, “factorial = 6”.

Finalmente se imprime por pantalla el resultado obtenido.

TIPO DE DATO: Entero Positivo

DATO: 3

RESULTADO OBTENIDO:

“El Factorial de 3 es: 6”

TIPO DE DATO: Entero Negativo

DATO: -3

RESULTADO OBTENIDO:

“No es un numero positivo”

“Ingrese un número entero positivo:”

EJERCICIO 11:

**Crear un algoritmo que muestre los primeros 10 números de la sucesión de Fibonacci. La sucesión comienza con los números 0 y 1 y, a partir de éstos, cada elemento es la suma de los dos números anteriores en la secuencia: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55...**

Para realizar este algoritmo se inicializa una variable “elemento” en 0 y una lista “Fibonacci” con sus dos primeros elementos “0;1”. Luego se utiliza una sentencia for con un rango que comienza en 1 y termina en 8 “for i in range (1,9)”. Dentro de esta, se hace la operación para conseguir los elementos de la sucesión:

“elemento = fibonacci[i-1] + fibonacci[i], ejemplo de la primera iteración:

elemento = 0, fibonacci[i-1] = 0 y fibonacci[i] = 1, por lo tanto, el tercer elemento de la sucesión es igual a la suma de los dos anteriores. Luego el elemento se agrega a la lista “Fibonacci” con el método append(). Y así sucesivamente con el resto de los elementos hasta salir de la sentencia for. Finalmente, se imprime por pantalla el resultado obtenido.

RESULTADO OBTENIDO:

“[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]”

EJERCICIO 12:

**Escribir un programa que permita al usuario ingresar 6 números enteros, que pueden ser positivos o negativos. Al finalizar, mostrar la sumatoria de los números negativos y el promedio de los positivos. No olvides que no es posible dividir por cero, por lo que es necesario evitar que el programa arroje un error si no se ingresaron números positivos.**

Para realizar este programa, en primer lugar, se definieron dos listas, una para guardar los números positivos y otra para guardar los números negativos. Estas son “numPositivos” y “numNegativos”. En segundo lugar, se definen las funciones para generar el promedio y la sumatoria de los números contenidos en las respectivas listas. “calcularPromedio(listaNumeros)“y “sumatoriaDenumeros (listaNumeros)”. Vale aclarar que, dentro de la función para calcular el promedio hay una condición que valida si la lista de números positivos existe, ya que no se puede dividir por cero. En tercer lugar, se pide el ingreso de los 6 números utilizando un ciclo for. Dentro de este se diferencian los números positivos y negativos utilizando dos condiciones y se agregan a la lista que les corresponde. Finalmente se muestra por pantalla los resultados obtenidos.

TIPO DE DATOS: Enteros

DATOS: 1;2;3;-1;-2;-3

RESULTADO OBTENIDO:

“---------------NUMEROS POSITIVOS INGRESADOS---------------

[1, 2, 3]

---------------NUMEROS NEGATIVOS INGRESADOS---------------

[-1, -2, -3]

------------------------PROMEDIO---------------------------

El Promedio de los números positivos es: 2.0

-----------------------------------------------------------

------------------------SUMATORIA--------------------------

La sumatoria de los números negativos es: -6

-----------------------------------------------------------

TIPO DE DATOS: Enteros

DATOS: -1;-2;-3;-4;-5;-6

RESULTADO OBTENIDO:

---------------NUMEROS POSITIVOS INGRESADOS---------------

[]

---------------NUMEROS NEGATIVOS INGRESADOS---------------

[-1, -2, -3, -4, -5, -6]

-----------------------------------------------------------

------------------------PROMEDIO---------------------------

Lista de números vacía, no se puede dividir por 0

El Promedio de los números positivos es: None

-----------------------------------------------------------

------------------------SUMATORIA--------------------------

La sumatoria de los números negativos es: -21

-----------------------------------------------------------

TIPO DE DATOS: Enteros

DATOS: 1;2;3;4;5;6

RESULTADO OBTENIDO:

---------------NUMEROS POSITIVOS INGRESADOS---------------

[1, 2, 3, 4, 5, 6]

---------------NUMEROS NEGATIVOS INGRESADOS---------------

[

-----------------------------------------------------------

------------------------PROMEDIO---------------------------

El Promedio de los números positivos es: 3.5

-----------------------------------------------------------

------------------------SUMATORIA--------------------------

La sumatoria de los números negativos es: 0

-----------------------------------------------------------