**PUNTO 1**

**Haga un resumen de conceptos mediante la lectura del artículo.**

***R/*** En Python, como en todos los lenguajes de programación, existen variables, las cuales son datos que se almacenan en la memoria de un computador con el fin de ser utilizadas a lo largo del programa. Estas variables se pueden utilizar en todo el programa, se les puede cambiar el valor en cualquier momento, entre otros, a excepción de las variables en las funciones.

Para que una función pueda trabajar con una variable fuera de esta, la función deberá ser definida con un parámetro y a la hora de ejecutarla, se deberá pasarle un argumento. Para que esta variable usada por la función sea útil para el resto del código se deberá retornar y almacenar su valor fuera de la función, por ejemplo:

a = 5

def cambiar\_a(*a*):

    a = 3

    return a

a = cambiar\_a(a)

print(a)

# Imprime 3

De lo contrario, obtenemos:

a = 5

def cambiar\_a ():

    a = 3

    print(a)

    return

cambiar\_a ()

print(a)

# Imprime 3 y luego imprime 5

De esta forma hemos creado “a” como una variable local dentro de la función cambiar\_a () la cual no tiene ninguna relación o afecto en la variable “a” definida en la primera línea. Pero, ¿Qué es una variable local?

Las variables en las funciones de Python pueden adquirir diferentes propiedades, siendo estas:

**Local:** Las variables locales sólo existen en la propia función y no son accesibles desde niveles superiores.

**Libre:** Son variables que no están definidas dentro de la función y no entran como argumentos al llamarla, pero Python detecta que esta variable no ha sido declarada y busca una declaración o valor inmediatamente un nivel por encima de la función, así:

a = 5

def cambiar\_a ():

    print(a)

    return

cambiar\_a ()

print(a)

# Imprime 5 y luego imprime 5

Si a la variable se le asigna valor en algún nivel intermedio la variable se considera no local y si se le asigna en el programa principal la variable se considera global. En este caso, es una variable libre global ya que la variable encuentra su declaración en el programa principal.

def funcion ():

    def sub\_funcion ():

        print(a)

        return

    a = 3

    sub\_funcion ()

    print(a)

    return

a = 4

funcion ()

print(a)

# Imprime 3, luego 3 y luego 4

En este caso la variable ***a*** en la función ***sub\_funcion ()*** es una variable libre No local ya que encuentra su definición inmediatamente en un nivel superior a ésta pero que no llega a ser el programa principal.

**Global:** Son variables que no se encuentran definidas dentro de la función pero que se declaran con la palabra reservada ***global*** para indicar que la variable tiene una definición fuera de la función y que ésta se podrá utilizar, además si se le hacen cambios a esta variable siendo global dentro de la función, los cambios se mantendrán aun estando fuera de la función.

def funcion ():

    global a

    print (a)

    a = 1

    return

a = 5

funcion ()

print (a)

# Imprime 5 y luego imprime 1

**No locales:** Son variables que no se encuentran definidas dentro de la función pero que se declaran con la palabra reservada ***nonlocal*** para indicar que la variable tiene una definición fuera de la función inmediatamente en un nivel superior, además si se le hacen cambios a esta variable siendo no local dentro de la función, los cambios se mantendrán aun estando fuera de la función.

def funcion ():

    def sub\_funcion ():

        nonlocal a

        print (a)

        a = 1

        return

    a = 3

    sub\_funcion ()

    print (a)

    return

a = 4

funcion ()

print (a)

# Imprime 3, luego 1 y luego imprime 4

**PUNTO 2**

**Basándose en lo realizado en el punto 1, conteste las siguientes preguntas:**

**¿Qué es el ámbito o alcance de una variable?**

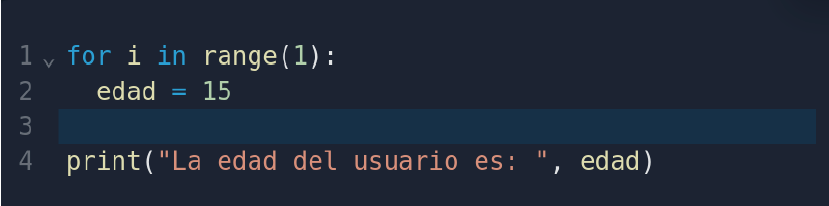
***R/*** El ámbito o alcance de una variable se define como la capacidad que tienen las variables de ser utilizadas en funciones sin una previa definición, sin ser pasada como argumento y que éstas puedan ser utilizadas o no en el resto de código sin necesidad de ser retornadas. Esto se puede hacer siempre y cuando las variables tengan el mismo nombre.

**Nombre los tipos de variables que se manejan en Python según su ámbito**

***R/***

* **Variables locales:** Variables definidas dentro de las funciones que no intervienen con otras variables con el mismo nombre fuera de ésta, para usarse en el resto de código debe ser retornada.
* **Variables libres:** Variables que no están definidas dentro de la función, pero Python busca una definición en los niveles superiores, es decir, en funciones por encima de ésta (variables libres no locales) o en el programa principal (variables libres globales).
* **Variables no locales:** Variables definidas con la palabra reservada ***nonlocal,*** estas variables no están declaradas en la función, pero si en un nivel inmediato encima de ella (si en este nivel tampoco se encuentra alguna declaración, el programa dará error), la alteración de esta variable será hecha para todo el programa, dentro o fuera de la función.
* **Variables globales:** Variables definidas con la palabra reservada ***global,*** estas variables no están declaradas en la función, pero si en un nivel superior o en el programa principal. La alteración de esta variable será hecha para todo el programa, dentro o fuera de la función.

**PUNTO 3**

**Analice el siguiente código y responda:**

**¿La variable edad debería generar algún error al ser usada en la línea 4? ¿sí?, ¿no?, argumente su respuesta en el contexto del ámbito de las variables en Python.**

***R/*** No, el tema de ámbito de las variables está enfocado al uso de las variables en funciones. En este caso, la variable será “global” y podrá ser utilizada en cualquier lugar del código siempre y cuando ésta este definida y no sea una función que la requiera (En este caso si la variable es utilizada con el mismo nombre dentro de la función, y es inmediatamente un nivel inferior al programa principal, Python tomará la variable como libre global).

**En Python, el alcance de una variable está definido por:**

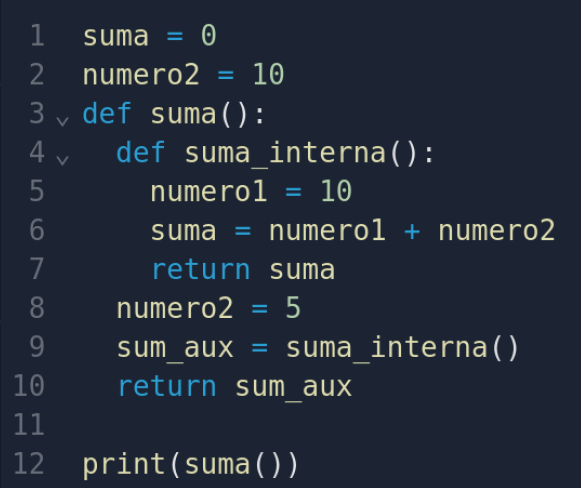
a) El programa principal y ciclos.

b) Condicionales y el programa principal.

***c) Funciones y el programa principal.***

d) Ciclos y funciones.

**PUNTO 4**

**Analice el siguiente código y desarrolle los siguientes puntos:**

**Usando la declaración global, haga que el retorno de la función suma\_interna () sea 20, de tal manera que dicha función use en su bloque de instrucciones la variable numero2 definida en la línea 2**

***R/***

suma = 0

numero2 = 10

def suma ():

    def suma\_interna ():

        numero1 = 10

        global numero2

        suma = numero1 + numero2

        return suma

    numero2 = 5

    sum\_aux = suma\_interna ()

    return sum\_aux

print (suma ())

**Clasifique como no local, local o global las siguientes variables:**

1. Variable suma usada en la línea 6

***R/*** Local

2. Variable numero2 definida en la línea 2

***R/*** Local

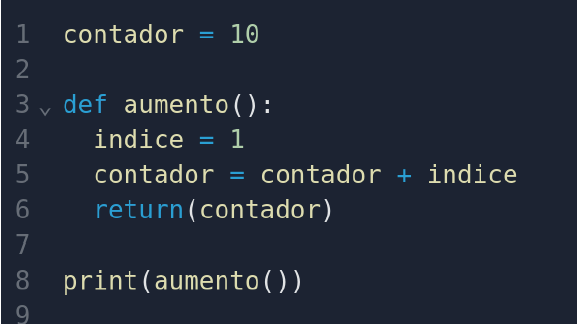
3. variable numero1 usada en la línea 6

***R/*** Local

4. variable numero2 usada en la línea 6

***R/*** No local y después del cambio, global

**PUNTO 5**

**Analice el siguiente código y desarrolle los siguientes puntos**

**Explique porque al ejecutar el programa ocurre un error crítico.**

***R/*** Porque en la línea 5 se está definiendo contador como contador + índice, lo que sucede es que, al estar redefiniendo la variable contador, se busca una declaración del valor de contador de manera local en la función o algún argumento que a esta se le pase, al ser inexistente la declaración del valor anteriormente mencionado, el programa se ejecuta y ocurre un error crítico de asignación de variables.

**Arregle el error que se presenta usando una de las siguientes declaraciones: nonlocal, global**

***R/*** El error se soluciona al definir entre la línea 4 y 5 contador como global, de esta forma contador de la línea 5 tomará el valor de contador en la línea 1.

contador = 10

def aumento ():

    indice = 1

    global contador

    contador = contador + indice

    return(contador)

print(aumento())