503203/503201/503215 Programación
Programación Nivel Certamen

Javier Vidal

6 de mayo de 2022

1.- **Secuencia de Fibonacci y Número Áureo** La secuencia de Fibonacci es aquella cuyo n-ésimo elemento f_n es igual a la suma de los dos anteriores ($f_{n-1} + f_{n-2}$, comenzando con $f_1 = 0$ y $f_2 = 1$). Por otra parte, el número áureo se define como $\phi = \frac{1+\sqrt{5}}{2}$, el cual se puede aproximar por la razón entre dos elementos consecutivos de la secuencia de Fibonacci, es decir, $\phi \approx \frac{f_{n+1}}{f_n}$, volviéndose más exacta a medida que aumenta n.

Desarrolle un algoritmo usando diagramas de flujo que calcule y despliegue el error que existe entre el número áureo exacto y su aproximación mediante la secuencia de Fibonacci para el elemento n.

Para efectos de cálculo de la raíz utilice el operador "**", esto es, $\sqrt{2}$ en Python se calcula como `2**0.5` y para el reporte del error utilice la función valor absoluto de Python, esto es, $| x |$ en Python se calcula como `abs(x)`.

Entradas: La única entrada al programa es un número entero positivo mayor o igual a 2, el cual debe ser validado. Si el número es mal ingresado se debe repetir la entrada hasta lograr un valor correcto para n

Salidas: La única salida del programa es el error de la aproximación.

Ejemplo de entrada 1: `5`

Ejemplo de salida 1: `0.04863`

Ejemplo de entrada 2: `9`

Ejemplo de salida 2: `0.0010136`

2.- **Inversión de un número entero** Una de las formas de determinar si un número m es *palíndrome* es comparando m con el propio número escrito al revés. Cabe señalar que "palíndrome" es un número (o palabra) que se lee igual de izquierda a derecha o de derecha a izquierda. Para ello se pide que escriba un programa en Python que lea un número m y que despliegue m pero escrito al revés.

Su programa solo debe utilizar operaciones numéricas (es decir, no se permite trabajar con strings) y debe guardar el número invertido en una variable antes de mostrarlo por pantalla.

Entradas: La única entrada al programa es un número entero m, $100 \leq m \leq 10^{12}$, el cual debe ser validado. Si el número es mal ingresado se debe repetir la entrada hasta lograr un valor correcto para m. Los ceros a la izquierda, si es que son ingresados, no se deben considerar en el cálculo del número invertido.

Salidas: La única salida del programa es m escrito al revés. Si el resultado tiene ceros a la izquierda deben ser omitidos.

Ejemplo de entrada 1: `123456789`

Ejemplo de salida 1: `987654321`

Ejemplo de entrada 2: `00500`

Ejemplo de salida 2: `5`

Ejemplo de entrada 2: `1092030`

Ejemplo de salida 2: `302901`

3.- **Terna de valor más alto** Obtener valores singulares como el mayor, el menor, el promedio o la media de un conjunto de datos es una tarea muy común en programación. En este problema Ud. tendrá que construir un programa Python que determine el valor más alto asociado a un conjunto de T ternas, cada una de las cuales está compuesta por tres números A, B y O. Los números A y B son valores reales (punto flotante) sin restricciones de rango. El número O sólo puede tomar un valor del conjunto {1,2,3,4} cada uno de los cuales representa a una de las respectivas operaciones $+$, $-$, $*$ o $/$. El programa tendrá que calcular el valor de cada terna (usando los valores de A, B y la operación O correspondiente) y seleccionar el valor más alto entre las T ternas.

Entradas: La entrada al algoritmo está compuesta por $3T + 1$ líneas. La primera contiene un número entero positivo correspondiente al valor de T, es decir, $T > 0$. Las siguientes T entradas alternarán dos valores reales A y B y un número entero O, $(1 \leq O \leq 4)$. Si el valor de O no cumple con el rango se debe ingresar nuevamente.

Salidas: La única salida del programa es el valor de la terna de mayor valor.

Ejemplo de entrada:

```
5
2.1   1.5   3
3.9   2.6   4
4.8  -3.0   1
8.6   2.5   4
2.1  -2.2   2
```

Ejemplo de salida: `4.3`

Observación: El resultado se obtiene a partir de los siguientes cálculos

```
2.1   1.5  3 representa 2.1 * 1.5 = 3.15
3.9   2.6  4 representa 3.9 / 2.6 = 1.5
4.8  -3.0  1 representa 4.8 + (-3.0) = 1.8
8.6   2.5  4 representa 8.6 / 2.5 = 3.44
2.1  -2.2  2 representa 2.1 - (-2.2) = 4.3
```

SOLUCIONES

Problema 1

```
# Secuencia de Fibonacci y Número Áureo

# Ingreso y validación de n
n=int(input('n='))
while n <= 1:
    n=int(input('Error, n debe ser positivo mayor o igual a 2.\nn='))

# Cálculo de f_n y de f_n+1
f_n=0
f_nmas1=1
if n>2:
    i=1
    while i<n:
        f_nmas2=f_nmas1+f_n
        f_n=f_nmas1
        f_nmas1=f_nmas2
        i=i+1
# print(f_n,f_nmas1)
# Cálculo de número áureo y aproximación de número áureo
phi=(1+5**0.5)/2
app_phi=f_nmas1/f_n
# print(phi, app_phi, abs(phi-app_phi))
print(abs(phi-app_phi))
```

Problema 2

```
# Inversión de número

# Ingreso y validación de m
m=int(input('m='))
while m < 100 or m>10**12:
    m=int(input('Error, m debe ser positivo mayor o igual a 2.\nm='))

# Inversión de m
aux=m
p=10
m_inv=0
while aux!=0:
    d=aux%10
    m_inv=m_inv*p+d
    aux=aux//10
print(m_inv)
```

Problema 3

```
# Terna de valor más alto
```

```
# Ingreso y validación de T
T=int(input('T='))
while T <= 0:
    T=int(input('Error, T debe ser positivo.\nT='))
# Lee primer valor para sacar el mayor de referencia
A=float(input('A='))
B=float(input('B='))
O=int(input('O='))
while O<1 or O>4:
    O=int(input('Error, O debe ser 1, 2 3 o 4.\nO='))
if O==1:
    valor_mayor=A+B
elif O==2:
    valor_mayor=A-B
elif O==3:
    valor_mayor=A*B
else:
    valor_mayor=A/B
for i in range(T-1):
    A=float(input('A='))
    B=float(input('B='))
    O=int(input('O='))
    while O<1 or O>4:
        O=int(input('Error, O debe ser 1, 2 3 o 4.\nO='))
    if O==1:
        valor=A+B
    elif O==2:
        valor=A-B
    elif O==3:
        valor=A*B
    else:
        valor=A/B
    if valor>valor_mayor:
        valor_mayor=valor
print(valor_mayor)
```