Algoritmos y Estructura de Datos I

Ejercitación Recursividad

1) Escriba una función recursiva que tenga un parámetro n de tipo entero y que devuelva el n-ésimo número de Fibonacci.

main.py

```python
from algo1 import *

def Fibonacci(n):
    if n==0 or n==1:
        return n
    return Fibonacci(n-1) + Fibonacci(n-2)


check=False
while check==False:
    n=input_int("Ingrese un número entero positivo: ")
    check=True
    if n<0:
        print("Debe ser un número positivo")
        check=False
print(f"Número {n} de Fibonacci: {Fibonacci(n)}")

```

Link Repl.it: https://replit.com/@Paulonia/Martinez13866RecursividadEj1#main.py

2) Escribir una función recursiva que devuelva la suma de los primeros N enteros. Se debe pedir la carga de N como un valor entero positivo.

main.py

```python
from algo1 import *

def suma(n):
    if n==1:
        return n
    return n+(suma(n-1))

check=False
while check==False:
    n=input_int("Ingrese un número entero positivo: ")
    if n>0:
        check=True
resultado=suma(n)
print(f"La suma de los primeros {n} números es: {resultado}")

```

Link Repl.it: https://replit.com/@Paulonia/Martinez13866RecursividadEj2#main.py

3) Escribir un programa que encuentre la suma de los enteros positivos pares desde N hasta 2. Chequear que si N es impar se imprima un mensaje de error.

main.py

```python
from algo1 import *

def sumaPares(n):
    if n==2:
        return n
    elif n%2==0:
        return n+(sumaPares(n-2))
    else:
        return print("Error")

check=False
while check==False:
    n=input_int("Ingrese un número entero positivo: ")
    if n>0:
        check=True
resultado=sumaPares(n)
if n%2==0:
    print(f"La suma de los pares desde 2 hasta {n} es: {resultado}")
```

Link Repl.it: https://replit.com/@Paulonia/Martinez13866RecursividadEj3#main.py

4) Escriba una función recursiva que ordene una lista enlazada de enteros de menor a mayor basándose en la siguiente idea: coloque el nodo del valor más pequeño en la primera ubicación y luego ordene el resto de la lista con una llamada recursiva.

main.py

```python
from algo1 import *
from mylinkedlist import *

def move(L,posInicio,posFinal):
  #Ordena la lista ingresada de menor a mayor elemento
  #Caso base
  if posFinal==length(L)-1:
    return
  #Mover el current a la posición de destino
  current=L.head
  if posFinal!=0:
    for i in range (0,posFinal):
      current=current.nextNode
  #Buscar valor más chico
  menor=current.value
  position=posFinal
  for i in range(posFinal,length(L)):
    if current.value<=menor:
      menor=current.value
      posInicio=position
    position+=1
    current=current.nextNode
  #Mover un nodo de un lugar(posInicio) a otro(posFinal)
  element=access(L,posInicio)
  current=L.head
  if posFinal!=0:
    for i in range (0,posFinal):
      current=current.nextNode
  if (posInicio!=posFinal):
    for i in range (posFinal,posInicio-1):
      current=current.nextNode
    current.nextNode=current.nextNode.nextNode
    insert(L,element,posFinal)
  return move(L,posInicio,posFinal+1)

L=LinkedList

llenarLista(L)
printList(L)
move(L,0,0)
print("Lista ordenada de menor a mayor: ")
printList(L)
```

Link Repl.it: https://replit.com/@Paulonia/Martinez13866RecursividadEj4#main.py

5) Toda función recursiva puede ser reimplementada utilizando un TAD PILA o STACK. Implementar el ejercicio 1 sin utilizar recursividad y haciendo uso únicamente de operaciones elementales provistas por el módulo my stack.py

main.py

```python
from algo1 import *
from mystack import *
from mylinkedlist import printList

def stack_fibonacci(num):
    S=LinkedList()
    push(S,0)
    push(S,1)
    resultado=0
    if num==1:
        return 1
    for i in range(0,num-1):
        aux1=pop(S)
        aux2=pop(S)
        resultado=aux1+aux2
        push(S,aux1)
        push(S,resultado)
    return resultado


check=False
while check==False:
    n=input_int("Ingrese un número entero positivo: ")
    if n>=0:
        check=True
print(f"El número {n} de Fibonacci es: {stack_fibonacci(n)}")

```

Link Repl.it: https://replit.com/@Paulonia/Martinez13866RecursividadEj5#main.py