Paula Martinez 13866

Algoritmos y Estructura de Datos I

Trabajo Práctico – Listas

1) Dadas las siguientes listas de elementos:

A) 24, 45, 3, 67, 89, 345, 54, 22, 3, 678

B) 46, 34, 64, 59, 12, 15, 234, 567, 12, 33

Implemente un único algoritmo en pseudo-python que permita resolver las siguientes consignas paso a paso:

a. Llenar dos listas A y B con los elementos dados, e imprimir su contenido

```
# - Ejercicio 1 -
A=LinkedList()
B=LinkedList()

add(A,678)
add(A,3)
add(A,22)
add(A,54)
add(A,345)
add(A,89)
add(A,67)
add(A,3)
add(A,45)
add(A,24)
print("Lista A: ")
printList(A)

add(B,33)
add(B,12)
add(B,567)
add(B,234)
add(B,15)
add(B,12)
add(B,59)
add(B,64)
add(B,34)
add(B,46)
print("Lista B: ")
printList(B)
```

b. Crear una lista C e intercalar los elementos de las dos listas anteriores, imprimir su contenido

```
# b)
C=LinkedList()
intercalarListas(A,B,C)
print("Lista C:")
printList(C)
```

```
def intercalarListas(A,B,C): # b)
  #Llenar una lista C con valores intercalados de las listas A y B
  currentA=A.head
  currentB=B.head
  size=length(A)+length(B)
  checkA=True
  for i in range (0,size):
    if (checkA==True):
      insert(C,currentA.value,i)
      currentA=currentA.nextNode
      checkA=False
    else:
      insert(C,currentB.value,i)
      currentB=currentB.nextNode
      checkA=True
```

c. Buscar en la lista C todos los elementos pares de la lista A y eliminarlos, imprimir la lista C final

```
# c)
eliminarPares(A,C)
print("Lista C sin elementos pares de la lista A: ")
printList(C)
```

```
def eliminarPares(L,L2): # c)
    #Eliminar de la segunda lista los elementos pares de la primer lista
    current=L.head
    while current!=None:
        if (current.value%2==0):
            delete(L2,current.value)
        current=current.nextNode
```

d. Generar una lista D con todos los elementos impares de C, finalmente imprimir el contenido de D.

```
# d)
D=LinkedList()
buscarImpares(C,D)
print("Lista D: ")
printList(D)
```

```
def buscarImpares(L,L2): # d)
    #Crear una lista con los elementos impares de L
    current=L.head
    cont=0
    while current!=None:
        if (current.value%2!=0):
            insert(L2,current.value,cont)
            cont+=1
        current=current.nextNode
```

e. Quitarle a la lista A los elementos repetidos y añadirle al final todos los elementos de B que se encuentren entre 50 y 100, imprimir la lista A resultante.

```
def sacarRepetidos(L): # e)
    #Sacar elementos repetidos de una lista
    current=L.head
    while current!=None:
        current2=L.head
        while current2!=None:
            if current!=current2 and current.value==current2.value:
                delete(L,current.value)
            current2=current2.nextNode
        current=current.nextNode

```

```
# e)
sacarRepetidos(A)
#añadirle al final todos los elementos de B que se encuentren entre 50 y 100
size=length(A)
current=B.head
while current!=None:
    if current.value>50 and current.value<100:
        insert(A,current.value,size)
        size+=1
    current=current.nextNode
print("Lista A cambiada: ")
printList(A)
print("")

```

2) Desarrolle un algoritmo en pseudocódigo que permita cargar una lista en donde su campo “value” sea igual a una estructura “Empleado” que tenga tres campos: “nombre”, “edad” y “nroLegajo”.

a. Cargar la lista de empleados

b. Imprimir la lista cargada

Empleados:
Eduardo Ángel, 34, 2
Juan Carlos, 23, 5
Luis Esteban, 32, 7
Juan Carlos, 23, 5
Pedro Augusto, 40, 9
Luis Esteban, 32, 7
Pedro César, 45, 8
Eduardo Ángel, 34, 2
Luis Esteban, 32, 7

```
149  # - Ejercicio 2 -
150  class LinkedListEmpleado:
151      head=None
152  class Empleado:
153      nombre=None
154      edad=None
155      nroLegajo=None
156  class NodeEmpleado:
157      value=Empleado
158      nextNode=None
159
160  listaEmpleados=LinkedListEmpleado()
161
162  dato=Empleado()
163  dato.nombre="Luis Esteban"
164  dato.edad=32
165  dato.nroLegajo=7
166  add(listaEmpleados,dato)
167
168  dato=Empleado()
169  dato.nombre="Eduardo Ángel"
170  dato.edad=34
171  dato.nroLegajo=2
172  add(listaEmpleados,dato)
173
174  dato=Empleado()
175  dato.nombre="Pedro César"
176  dato.edad=45
177  dato.nroLegajo=8
178  add(listaEmpleados,dato)
179
180  dato=Empleado()
181  dato.nombre="Luis Esteban"
182  dato.edad=32
183  dato.nroLegajo=7
184  add(listaEmpleados,dato)
185
186  dato=Empleado()
187  dato.nombre="Pedro Augusto"
188  dato.edad=40
189  dato.nroLegajo=9
190  add(listaEmpleados,dato)
```

```
191
192 dato=Empleado()
193 dato.nombre="Juan Carlos"
194 dato.edad=23
195 dato.nroLegajo=5
196 add(listaEmpleados,dato)
197
198 dato=Empleado()
199 dato.nombre="Luis Esteban"
200 dato.edad=32
201 dato.nroLegajo=7
202 add(listaEmpleados,dato)
203
204 dato=Empleado()
205 dato.nombre="Juan Carlos"
206 dato.edad=23
207 dato.nroLegajo=5
208 add(listaEmpleados,dato)
209
210 dato=Empleado()
211 dato.nombre="Eduardo Ángel"
212 dato.edad=34
213 dato.nroLegajo=2
214 add(listaEmpleados,dato)
215
216 printListEmpleados(listaEmpleados)
217 print("")
218
```

```
51 def printListEmpleados(L):
52   #Imprimir lista de empleados
53   current=L.head
54   while current!=None:
55     print(current.value.nombre, end=", ")
56     print(current.value.edad, end=", ")
57     print(current.value.nroLegajo)
58     current=current.nextNode
59
```

3) Para el ejercicio anterior:

a. Eliminar los elementos donde "nroLegajo" se encuentren duplicados.

```
219 # - Ejercicio 3 -
220 # a)
221 sacarLegajosDuplicados(listaEmpleados)
222 print("No duplicados")
223 printListEmpleados(listaEmpleados)
224
```

```
61 def sacarLegajosDuplicados(L): # a)
62   #Sacar los numeros de legajos duplicados de una lista de empleados
63   current=L.head
64   while current!=None:
65     current2=L.head
66     while current2!=None:
67       if current!=current2 and current.value.nroLegajo==current2.value.nroLegajo:
68         delete(L,current2.value)
69       current2=current2.nextNode
70     current=current.nextNode
71
```

b. Agregar antes del legajo número 7 el siguiente: Ernesto Andrés, 55, 6

```
# b)
nuevoEmpleado=Empleado()
nuevoEmpleado.nombre="Ernesto Andrés"
nuevoEmpleado.edad=55
nuevoEmpleado.nroLegajo=6
current=listaEmpleados.head
pos=0
while current!=None:
    if current.value.nroLegajo==7:
        insert(listaEmpleados,nuevoEmpleado,pos)
    pos+=1
    current=current.nextNode
print("")
print("Lista con nuevo empleado: ")
printListEmpleados(listaEmpleados)

```

c. Mover el legajo 9 luego del legajo 8

```
# c)
current=listaEmpleados.head
while current!=None:
    if current.value.nroLegajo==9:
        aux=current.value
        delete(listaEmpleados,current.value)
    current=current.nextNode
current=listaEmpleados.head
pos=0
while current!=None:
    if current.value.nroLegajo==8:
        insert(listaEmpleados,aux,pos+1)
    pos+=1
    current=current.nextNode
```

d. Imprimir la lista resultante

```
print("Lista resultante:")
printListEmpleados(listaEmpleados)

```

4) Desarrolle un algoritmo que ordene de manera inversa los elementos (nodos) de una lista.

```
# - Ejercicio 4 -
def invertirLista(L):
    #ordenar de manera inversa los elementos (nodos) de una lista
    newL=LinkedList()
    size=length(L)
    current=L.head
    for i in range(0,size):
        aux=current.value
        add(newL,aux)
        current=current.nextNode
        delete(L,aux)
    current=newL.head
    for i in range(0,size):
        insert(L,current.value,i)
        current=current.nextNode
```

Link Repl.it:
https://replit.com/@Paulonia/Martinez13866TPListas#main.py