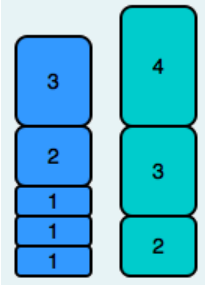
Comenzado en	Friday, 25 de August de 2023, 17:22
Estado	Terminados
Finalizado en	Friday, 25 de August de 2023, 18:33
Tiempo empleado	1 hora 11 mins
Calificación	**74.00** de un total de 100.00

Pregunta 1

Completada

Puntúa 9.00 sobre 35.00

Cuenta con dos pilas como se muestra a continuación. El elemento que guarda cada pila es un entero que corresponde a la edad de un niño en un rango de 1-5 años.



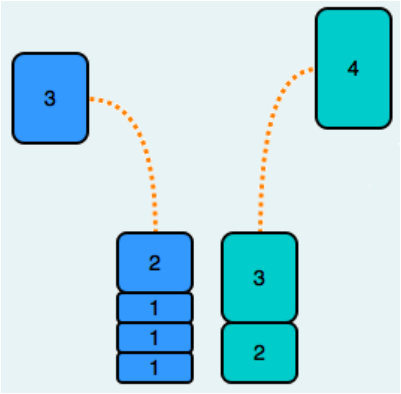
En este sentido la pila de la izquierda tiene 3 registros con niños de 1 año, un niño de 2 años y en la cima se encuentra alguien con 3 años. El algoritmo que debe diseñar debe nivelar ambas pilas para que la suma de las edades en cada una sea igual.

Por ejemplo:

```
pila_1: 1 -> 1 -> 1 -> 2 -> 3
suma_pila_1: 8

pila_2: 2 -> 3 -> 4
suma_pila_2: 9
```

Para resolver este problema la solución sería de la siguiente manera:



Como se observa se han extraído los primeros nodos de cada pila y con esto el resultado es el siguiente:

```
pila_1: 1 -> 1 -> 1 -> 2
suma_pila_1: 5

pila_2: 2 -> 3
suma_pila_2: 5
```

Tome en cuenta los siguientes aspectos:

1. Construya la clase ***Pila***, tomando en cuenta que sus elementos serán valores enteros (en este caso vendrían siendo las edades).
2. Elabore la función ***Balancear*** que reciba de parámetro a ambas pilas. Esta función debe determinar si es posible el balanceo en función de la sumatoria de las edades, de ser así, debe devolver cada pila resultante, de lo contrario se deberá indicar con un mensaje "Balanceo imposible de llevar a cabo", quiere decir que su algoritmo debe funcionar para cualquier escenario.
3. Construya dos instancias de la clase ***Pila*** con los valores iniciales como se ve en la primer imagen.
4. Llame la función ***Balancear*** y devuelva el resultado.
5. Luego de haber probado que el código funciona correctamente copie y pegue en el cuadro que aparece a continuación.

```
class Pila:
    def __init__(self):
        self.pila = []

    def apilar(self, valor):
        self.pila.append(valor)

    def desapilar(self):
        if self.pila:
            return self.pila.pop()

    def sumatoria(self):
        return sum(self.pila)

def Balancear(pila1, pila2):
```

Comentario:
- La lógica es incorrecta para realizar el balanceo.

- Se colocan algunos puntos por el esfuerzo.

Pregunta 2

Completada

Puntúa 6.00 sobre 6.00

Si se intenta desreferenciar un apuntador que contiene una dirección inválida pueden ocurrir cosas como las siguientes (seleccione las que correspondan):

- [x] a. El sistema operativo detecta el acceso inadecuado a una dirección de memoria, en cuyo caso detiene abruptamente el programa.
- [x] b. La dirección esta fuera de la zona de memoria utilizada para almacenar datos y más bien esta en la zona donde se almacenan las instrucciones del programa.
- [ ] c. Se libera la memoria reservada en esa dirección, ya que la variable asociada terminó su ámbito.
- [x] d. Se obtiene un valor incorrecto en una o más variables debido a que no fue debidamente inicializada la zona de memoria.

Su respuesta es correcta.

Las respuestas correctas son:
Se obtiene un valor incorrecto en una o más variables debido a que no fue debidamente inicializada la zona de memoria.,

El sistema operativo detecta el acceso inadecuado a una dirección de memoria, en cuyo caso detiene abruptamente el programa.,

La dirección esta fuera de la zona de memoria utilizada para almacenar datos y más bien esta en la zona donde se almacenan las instrucciones del programa.

Pregunta 3

Completada

Puntúa 6.00 sobre 6.00

Durante la ejecución de un programa puede variar el tamaño de una región o, incluso, pueden crearse nuevas o eliminarse regiones existentes. El sistema de memoria debe controlar qué regiones están presentes en el mapa de memoria y cuál es el tamaño actual de cada una. Además de facilitar la depuración, se elimina la necesidad de reservar memoria física para zonas del mapa que no estén asignadas al proceso en un determinado instante. El sistema operativo debe almacenar por cada proceso una tabla de regiones que indique las características de las regiones del proceso (tamaño, protección, etc.).

¿Qué característica se ha definido en el contexto de gestión de memoria?

- a. Proporcionar a los procesos mapas de memoria muy grandes.
- b. Permitir que los procesos compartan memoria.
- c. Proporcionar protección entre los procesos.
- d. Dar soporte a las distintas regiones del proceso.
- e. Maximizar el rendimiento del sistema.

Su respuesta es correcta.

La respuesta correcta es:
Dar soporte a las distintas regiones del proceso.

Pregunta 4

Completada

Puntúa 7.00 sobre 7.00

Seleccione aquellas características que describen el método de paginación:

- a. Puede conducir a una fragmentación externa.
- b. Agrupación lógica de la información en bloques de tamaño variable.
- c. Se establece una tabla de páginas para trasladar las direcciones lógicas a físicas.
- d. Se divide la memoria física en bloques de tamaño fijo llamados marcos.
- e. Puede conducir a una fragmentación interna.
- f. Con este método, un programa puede ocupar más de una partición y éstas no tienen por qué estas contiguas.

Su respuesta es correcta.

Las respuestas correctas son:
Se divide la memoria física en bloques de tamaño fijo llamados marcos.,

Se establece una tabla de páginas para trasladar las direcciones lógicas a físicas.,

Puede conducir a una fragmentación interna.

Pregunta 5

Completada

Puntúa 6.00 sobre 6.00

No es más que un caché especial que se utiliza para llevar un seguimiento de las transacciones utilizadas recientemente. Contiene las entradas de la tabla de páginas que se han utilizado. Dada una dirección virtual, esta se busca y si se encuentra, se recupera el número de trama y se forma la dirección real.

a. Segmentación
b. Memoria virtual
c. TLB
d. Swapping
e. Paginación

Su respuesta es correcta.

La respuesta correcta es:
TLB

Pregunta 6

Completada

Puntúa 40.00 sobre 40.00

Dadas dos listas simples enlazadas desarrolle una función para vincularlas y ordenar sus elementos:

Ejemplo:

```
Lista1: 7 -> 9 -> 11 -> 17
Lista2: 11 -> 12 -> 14
```

El resultado luego de combinar y ordenar ambas listas deberá ser:

```
ListaResultado: 7 -> 9 -> 11 -> 11 -> 12 -> 14 -> 17
```

Tome en cuenta los siguientes aspectos:

1. Construya la clase ***Nodo*** que este definida con un valor entero.
2. Construya la clase ***ListaEnlazada***.
3. En la clase ***ListaEnlazada*** construya la función ***CombinarListas*** que reciba como parámetros las listas que se vincularan.
4. Además construya la función ***Ordenar*** que organice todos los elementos, tome en cuenta que si encuentra repetidos también lo debe tomar en cuenta.
5. Por último construya la función ***Recorrer*** que imprima todos los elementos de la lista resultante (ordenada).
6. Luego de haber probado que el código funciona correctamente copie y pegue en el cuadro que aparece a continuación.

```
class Nodo:
    def __init__(self, valor):
        self.valor = valor
        self.siguiente = None

class ListaEnlazada:
    def __init__(self):
        self.primero = None

    def insertar(self, valor):
        nuevo_nodo = Nodo(valor)
        if self.primero is None:
            self.primero = nuevo_nodo
        else:
            actual = self.primero
```

Comentario:
Excelente!

◄ Hoja 8

Ir a...

Parcial 2 ►