Introducción a la Algorítmica y Programación

## Trabajo Práctico Integrador

## Integrantes

* Augusto Lovagnini
* Giovanni Buchieri
* Genaro Pennone

**Actividad**

**Ej. 15)** Para administrar una lista de personas se requiere hacer diversas acciones que permitan ese trabajo. Para almacenar los nombres se utilizará un registro que contiene dos campos, uno con un arreglo donde se guardan los nombres y otro con la cantidad actual de nombres almacenados en el arreglo (Utilice un registro con un campo arreglo y otro campo con la cantidad de datos que se han cargado al arreglo). La cantidad máxima que se podrá almacenar será de 1000 nombres. Para poder administrar la lista se deben desarrollar las siguientes acciones y/o funciones:

a) Una función llamada Vacia que reciba como parámetro el registro (con el arreglo de nombres y la cantidad de nombres cargados), y devuelva verdadero si la lista está vacía y sino debe devolver falso.

b) Una función llamada Llena que reciba como parámetro el registro (con el arreglo de nombres y la cantidad de nombres cargados), y devuelva Verdadero si el arreglo está completamente lleno (es decir si ya tiene 1000 nombres cargados) y sino debe devolver Falso.

c) Una acción que permita insertar un nombre en el arreglo. Para ello se pasará como parámetros el registro (con el arreglo de nombres y la cantidad de nombres cargados), y el nuevo nombre a insertar. El nuevo nombre se inserta siempre al final de la lista. Después de ejecutada la acción, la cantidad debe quedar incrementada en una unidad.

d) Una acción que permita suprimir al primer nombre de la lista, no importa cuál es. Simplemente suprime el primero cada vez que la acción es ejecutada. El parámetro que debe recibir es el registro (con el arreglo de nombres y la cantidad de nombres cargados). Después de ejecutada la acción, la cantidad debe quedar decrementada en una unidad.

e) Una acción que permita mostrar todos los nombres de la lista. Los parámetros que debe recibir son: el arreglo y la cantidad de nombres que tiene cargados.

## Análisis

**Datos**: listaNombres, nuevoNombre, selección, volver //corregir si es que NuevoNombre, selección y volver van como datos.

**Resultados**: arreglo, msg

**Relaciones y Subproblemas:**

salir = falso

Usaremos la estructura iterativa “Mientras” con la condición de continuación “salir = falso”, esto quiere decir que hasta que la variable lógica “Salir” no sea verdadera se continuara ciclando.

Dentro de esta estructura:

Primero se mostrará un menú el cual le preguntará al usuario que quiere hacer, entre las opciones esta:

* Insertar Nuevo Nombre
* Eliminar Primer Nombre de la lista
* Mostrar la lista
* Salir

Luego se pedirá la entrada de la variable “Selección” en la cual se va poder ingresar un valor del 1 al 4, ya que son 4 opciones las cuales el usuario puede elegir dentro del programa, para garantizar que es un numero entre el 1 y el 4, utilizaremos de nuevo la estructura iterativa “Mientras”, esta hará que se siga pidiendo la entrada de la variable hasta que ingrese algún numero valido.

Usaremos la estructura “según” con las siguientes condiciones:

Cuando selección sea 1 (selección = 1): se ejecutará “CargarNuevoNombre(listaNombres)”

//Esta acción permite cargar un nuevo nombre en el arreglo, primero se fija que el arreglo no esté lleno, mediante la función “Llena” la cual devuelve verdadero si la función ya contiene 1000 nombres cargados o falso en caso no los tenga.

Cuando selección sea 1 (selección = 2): se ejecutará “SuprimirNombre(listaNombres)”

//Esta acción permite suprimir el primer nombre del arreglo. Si el arreglo este vacío no se podrá eliminar nada y mostrará un mensaje aclarando que no existe ningún nombre cargado, para saber si esta vacío, esta acción utiliza la función “Vacía” la cual devuelve verdadero si la función no tiene ningún nombre cargado.

Cuando selección sea 1 (selección = 3): se ejecutará “Mostrar(listaNombres)”

//Esta acción muestra en pantalla todos los nombres previamente ingresados, en caso que este vacío (utilizando la acción “Vacía” previamente mencionada), se mostrara un mensaje diciendo que no se ha encontrado ningún nombre.

Cuando selección sea 1 (selección = 4): a la variable salir se le asignara verdadero (salir ← verdadero)

//Esto va hacer que se termine el programa automáticamente.

Luego de ejecutar el condicional, se le preguntara al usuario (en caso de que haya seleccionado una opción entre la 1 y 3) si quiere volver al menú del inicio o salir del programa. Cuando inserte cualquier número (Excepto el 1), se volverá a ejecutar el menú. Cuando inserte 1 a la variable “salir” se le asignara verdadero (salir ← verdadero). Y Terminara el programa

## Diseño

**Algoritmo**

**Léxico**

Max = 1000

TElem = Cadena

TNombres= arreglo[1.. Max] de TElem

TData = < nombres ∈ TNombres, cant ∈ (0..Max) >

listaNombres ∈ TData

msg ∈ Cadena

salir ∈ Lógico

selección ∈ (1..4) //Subrango Enteros

volver ∈ Z

*Funcion* Vacia(dato nom ∈ TData) → Lógico

*Inicio*

Si(nom.cant = 0 ) entonces

← verdadero

Sino

← falso

fsi

*ffuncion*

*Funcion* Llena(dato nom ∈ TData)→ Lógico

*Inicio*

Si (nom.cant = 1000 ) entonces

← verdadero

Sino

← falso

fsi

*ffuncion*

*Acción* cargarNuevoNombre (dato-resultado nom ∈ TData)

*Léxico Local*

nuevoNombre ∈ Caracter

msg ∈ Cadena

*Inicio*

Si(Llena(nom)) entonces

msg ← “No es posible insertar el nuevo nombre ya que el arreglo está lleno”

salida: msg

sino

//Ingresa el Nuevo Nombre que quieres insertar

Entrada: nuevoNombre

nom.nombres[nom.cant+1] ← nuevoNombre

nom.cant ← nom.cant + 1

msg ← “Nuevo nombre cargado con éxito”

salida: msg

fsi

*facción*

*Acción* SuprimirNombre(dato-resultado nom ∈ TData)

Léxico Local

msg ∈ cadena

*Inicio*

Si (Vacia(arreglo)) entonces

Msg ← “No se pudo suprimir ningún nombre, porque el arreglo está vacío”

Salida: msg

Sino

nom.nombres[1] ← nom.nombres[nom.cant]

nom.cant ← nom.cant – 1

fsi

*facción*

*Accion* Mostrar (dato nom ∈ TData)

*Léxico Local*

msg ← Cadena

*Inicio*

Si(Vacio(nom)) entonces

msg ← “el arreglo está vacío”

sino

Para(i ← 1, i <= nom.cant, i ← i + 1) hacer

Salida: nom.nombres[i]

fpara

*faccion* **INICIO**  
salir ← falso

Mientras (salir = falso) hacer

msg ← "Que quieres hacer?

[1]Cargar Nuevo Nombre.

[2] Suprimir el primer Nombre.

[3] Mostrar lo que contiene el arreglo

[4] Salir

\*INGRESA AQUI EL NUMERO\*: "

Salida: msg

Entrada: selección

Según

(selección = 1): CargarNuevoNombre(listaNombres)

(selección = 2): SuprimirNombre(listaNombres)

(selección = 3): Mostrar(listaNombres)

(selección = 4): salir ← verdadero

Fsegun

Si (selección <> 4) entonces

msg ← “Si quiere SALIR presione [1], para volver al MENU Presione [Cualquier Otro Numero]”

salida: msg

Entrada: volver

Si(volver = 1) entonces

salir ← verdadero

fsi

fsi

fmientras

**FIN**

**Pruebas:**

Ejemplo: Se quiere Cargar una lista de 4 nombres: Juan, Pedro, Miguel, Luna

|  |  |  |  |
| --- | --- | --- | --- |
| CargarArreglo | | | |
| Ciclo | i | nom.cant | i <= nom.cant |
| 0 | - | 4 | - |
| 1 | 1 | 4 | V |
| 2 | 2 | 4 | V |
| 3 | 3 | 4 | V |
| 4 | 4 | 4 | V |
| 5 | 5 | 4 | F |

Ya ingresados estos 4 nombres, el usuario quiere insertar 3 mas (José, Martina, Lucas)

|  |  |  |
| --- | --- | --- |
| Llena(nom) | nom.cant | nuevoNombre |
| F | 4 | … |
| F | 5 | José |
| F | 6 | Martina |
| F | 7 | Lucas |

Ahora el usuario decide quitar el primer nombre de la lista:

Ejemplo antes de quitar el primer nombre: Ejemplo después de quitarlo:

|  |  |  |
| --- | --- | --- |
| Vacia(nom) | nom.cant | Lista de Nombres |
| F | 1 | Juan |
| F | 2 | Pedro |
| F | 3 | Miguel |
| F | 4 | Luna |
| F | 5 | José |
| F | 6 | Martina |
| F | 7 | Lucas |

|  |  |  |
| --- | --- | --- |
| Vacia(nom) | nom.cant | Lista de Nombres |
| F | 1 | Lucas |
| F | 2 | Pedro |
| F | 3 | Miguel |
| F | 4 | Luna |
| F | 5 | José |
| F | 6 | Martina |

El Usuario quiere que se muestren los nombres de la lista:

|  |  |  |  |
| --- | --- | --- | --- |
| Mostrar | | | |
| Ciclo | nom.cant | i | i <= nom.cant |
| 0 | 6 | - | - |
| 1 | 6 | 1 | V |
| 2 | 6 | 2 | V |
| 3 | 6 | 3 | V |
| 4 | 6 | 4 | V |
| 5 | 6 | 5 | V |
| 6 | 6 | 6 | V |
| 7 | 6 | 7 | F |