UTN – FRBA – Algoritmos y Estructura de Datos – Examen Final – 26/02/2017

Apellido y nombre:_____ Legajo: _____ Cursó con Prof: _____

Cantidad de hojas entregadas:_____ Nota:_____ Evaluó Prof: _____

LIFO y FIFO y sus características

Temas evaluados: Abstracción, estructuras de datos indexadas y enlazadas, resolución de problemas.

Definición del contexto

Como sabe las estructuras con criterio *FIFO* (primero en entrar, primero en salir) se implementan agregando al final un nuevo elemento y sacando del tope cuando se requiere recuperar un elemento. Si bien lo habitual es implementarlas mediante asignación dinámica en memoria, también es posible simular una implementación de este tipo en posiciones contiguas (vectores). La forma más elemental, si los elementos se agregan de uno por vez y si no se sobrepasa la capacidad máxima definida de ese vector, es agregar en el índice inmediato siguiente al último cargado, actualizando esta posición. Para retirar debería hacerse desde el comienzo de la estructura, también teniendo actualizada esta posición. En este caso frente y fin serán solo los índices de las posiciones de donde retirar (frente) o de donde agregar (fin). Algo similar puede hacerse con pilas, hasta incluso con listas, aunque en este caso la implementación puede ser un poco más compleja.

Dinámica del proceso para la implementación.

Se busca cargar los datos en dos estructuras simultáneamente, una que conserve el orden de ingreso (FIFO) y otra que lo invierta (LIFO). Una vez cargadas estas estructuras desarrollar una aplicación simple que deje en claro las particularidades del orden y el acceso al dato según el tipo de estructura. El recorrido dejara evidenciado que en una estructura los datos están leídos de izquierda a derecha y en la otra de derecha a izquierda

Problema

Para cumplir con lo que se propone, se requiere:

1. Leer de un archivo un conjunto de caracteres, **uno por vez**, controlando que haya caracteres y que no se supere el límite de la estructura contigua. Almacenarlos en una estructura contigua con criterio ***FIFO*** y, simultáneamente, en otra con criterio ***LIFO***. Para lectura por carácter en C ver al pie
2. Una vez cargadas ambas estructuras se debe determinar si el conjunto de caracteres ingresados forman un palíndromo, esto es una palabra o frase que es igual si es leída de izquierda a derecha o de derecha a izquierda. Ejemplo NEUQUEN, AMAN A PANAMA (ayuda: preste atención a los espacios en blanco).

Se pide

1. (1 puntos) **Codifique** la **declaración** todas estructuras de datos y las constantes que correspondan para la resolución correcta del problema, e inicialícelas adecuadamente.
2. (2 puntos)**Codifique o diagrame las funciones** *pop y push* para implementar una pila con asignación dinámica en memoria.
3. (2 puntos) **Codifique** o **diagrame la función** que recibe un vector de caracteres, un carácter a cargar en el mismo y un valor donde debe ser insertado. *agregarA_FIFO(vector, carácter,índice)*
4. (2 puntos) **Codifique** o **diagrame la función** que recibe el vector de caracteres el tope de donde tomar el carácter -el mismo debe ser actualizado- y retorna el carácter que estaba en el tope.

   suprimirDe_FIFO(vector, índice)
5. (1 puntos) **Codifique** o **diagrame la función** *ingresarCaracteres* que toma los caracteres desde un archivo y los almacene en las estructuras *LIFO y FIFO*, según la implementación y **uno por vez**.
6. (2 puntos) **Codifique** o **diagrame la función** *esPalíndromo* que reciba las estructuras adecuadas y determine si el conjunto de caracteres del flujo forman palíndromos.

Criterio de evaluación

La nota mínima para la aprobación es 6 (seis). Es condición requerida para la aprobación, además de la nota, que las definiciones de tipos y al menos una función deben estar resueltas correctamente.

1		2		3		4		5		6		Total
P	O	P	O	P	O	P	O	P	O	P	O	10
1		2		2		2		1		2		

Ayuda int fgetc(FILE *); Lee el siguiente carácter (si está presente) y avanza el indicador de posición. Si está en el final, indicador del final de archivo activado y fgetc retorna EOF. Es decir puede leer while (c=fgetc(f)) si hay carácter lo contiene en c. Retorna(hay carácter a leer? El carácter: EOF

1. (1 puntos) **Codifique** la **declaración** todas estructuras de datos y las constantes que correspondan para la resolución correcta del problema, e inicialícelas adecuadamente.

   a. Las constantes

   #define N 100 // cada invocación se N será reemplazada por el valor, en este caso 100

   b. Para la pila

```
struct Nodo{
        char info;          // el campo de la información es un carácter
        Nodo* sgte;         // al se una estructura auto referenciada el sgte es un puntero
};Nodo*pila = NULL;          // la pila inicializada como variable global
```

   c. Para la cola

```
char v[N];                  //simplemente un vector de char de N componentes
int inicio = 0; int fin = 0;    // comienzo y fin de la cola como índices inicializadoa
```

   d. Para el archivo

```
FILE * f = fopen("archivo", "rt");
```

2. (2 puntos)**Codifique o diagrame las funciones** *pop y push* para implementar una pila con asignación dinámica en memoria.

void push(Nodo*& pila, char x)	char pop (Nodo*& pila)

{ Nodo* p = new Nodo();
p->info = x;
p->sgte = pila;
pila = p;
return;
}
	{ Nodo* p = pila;
	char x = p->info;
	pila = p->sgte ;
	delete p;
	return x;
	}

3. (2 puntos) **Codifique** o **diagrame la función** cuyo encabezado es *void queue(char V[], char c, int* &i), que recibe un vector de caracteres, un carácter a cargar en el mismo y un valor fin donde debe ser insertado.

```
void agregarA_FIFO( char v[ ], char c, int &fin){
        v[fin] = c;
        fin++; //agrega al final, es el fin el que se modifica con cada ingreso
 return;
}
```

El control de no superar el tamaño fijo se hace en la función *ingresar Caracteres*

4. (2 puntos) **Codifique** o **diagrame la función** cuyo encabezado es *char unQueue(char V[], char c, int* &i), que recibe el vector de caracteres el tope de donde tomar el carácter -el mismo debe ser actualizado- y retorna el carácter que estaba en el tope

```
char suprimirDe_FIFO(char v[ ], int& inicio){
        char c = v[i];
        inicio++;// saca del inicio, en este caso el inicio es el índice que cambia
        return c;
}
```

5. (1 puntos) **Codifique** o **diagrame la función** *ingresarCaracteres* que toma los caracteres desde un archivo y los almacene en las estructuras *LIFO y FIFO*, según la implementación y de uno por vez.

```
void ingresarCaracteres(FILE* f, char v[ ], Nodo* &pila){
        int i = 0;
        int cantidadCaracteres = 0;
        char c;
        while( cantidadCaracteres<N && c = fgetc(f)){
                push(pila, c);
                queue(v,c,i);
                cantidadCaracteres++; //control para no superar tamaño estructura contigua
        };
        return;
        }
```

6. (2 puntos) **Codifique** o **diagrame la función** *esPalíndromo* que reciba las estructuras adecuadas y determine si el conjunto de caracteres del flujo forman palíndromos.

```
int esPalindromo(char v[], Nodo*& pila){
        int i = 0;
          while (pila)
                if(pop(pila) != unQueue(v,i)) retrun 0;
          return 1;
        }
```

Retorna (si es palíndromo? 1 : 0)