Parcial-C-Abril2020.pdf

Pv6lx

Programación de Sistemas y Concurrencia

2º Grado en Ingeniería Informática

Escuela Técnica Superior de Ingeniería Informática
Universidad de Málaga

UNIVERSIDAD DE MÁLAGA | uma.es

Dpto. de Lenguajes y Ciencias de la Computación

Programación de Sistemas y Concurrencia

Primer Control C
Curso 2019-2020

APELLIDOS_____NOMBRE_____

DNI _____ ORDENADOR _____GRUPO/TITULACIÓN_____

VERSIÓN 2. Bloque de C-Ejercicio Polinomios-Primera Parte

Un polinomio del tipo $2\,x^5 + \,3\,x^3 + 5x + 9$ podría representarse mediante una lista enlazada del modo siguiente:

Cada nodo contiene un monomio del polinomio, con su **coeficiente**, su **exponente**, y un enlace al monomio siguiente. Como puedes observar en el dibujo los monomios están ordenados según su exponente del monomio con exponente mayor ($2x^5$) al de menor (9). Observa, asimismo, que los monomios que tienen coeficiente 0 **están** almacenados en la lista (por ejemplo, el monomio de exponente 4 aparece en la lista con su coeficiente 0), aunque no hay ningún monomio con coeficiente 0 de exponente mayor que el grado del polinomio.

En esta primera parte del examen, debes implementar las siguientes funciones en el fichero fuente Polinomio.c. Se proporciona un fichero Principal.c para que puedas probar las funciones que vas implementando.

```
/*Crea el polinomio 0 (es decir, un polinomio con un solo monomio que tenga
exponente 0 y coeficiente 0).
*/
void polinomioCero(TPolinomio *p);

/*Devuelve el grado del polinomio, es decir, el mayor exponente de los
monomios que no son nulos. En el ejemplo, el grado es 5.
*/
unsigned int grado(TPolinomio p);

/* Devuelve el coeficiente del exponente exp del polinomio p.
*/
unsigned int coeficiente(TPolinomio p,unsigned int exp);

/* Inserta un monomio al principio del polinomio, solo si su coeficiente es
distinto de cero. Suponemos que esta función se llamará siempre con un
exponente exp mayor al grado actual del polinomio. No es necesario
comprobarlo. Además del monomio que se pide habrá que añadir todos los
monomios con coeficientes 0 necesarios para que el polinomio contenga todos
sus monomios. Por ejemplo, si se quiere añadir el monomio 3*x^4 y el grado
```

```
actual del polinomio es 1, habrá que añadir también los monomios 0*x^3 y
0*x^2.
*/
void insertarPrincipio(TPolinomio *p,unsigned int coef, unsigned int exp);

/* Inserta el monomio con coeficiente coef, y exponente exp en el polinomio
p, de manera que el polinomio quede ordenado. Si exp es mayor al grado actual
del polinomio entonces se insertará el monomio al principio, de acuerdo a lo
indicado para la función insertarPrincipio implementada anteriormente. En
otro caso, el monomio ya existe en el polinomio y solo habrá que actualizar
su coeficiente. El coeficiente y el exponente son números naturales (enteros
no negativo).
*/
void insertar(TPolinomio *p, unsigned int coef, unsigned int exp);

/*Escribe por la pantalla el polinomio con un formato similar al siguiente:
 [(2,5)(0,4)(3,3)(0,2)(5,1)(0,9)] para el polinomio de ejemplo. Observa que
se muestran todos los monomios, incluyendo los de coeficientes 0.
*/
void imprimir(TPolinomio p);

/* Elimina todos los monomios del polinomio, haciendo que la estructura se
quede vacía.
*/
void destruir(TPolinomio *p);
```

Ejercicio Polinomios-Segunda Parte

Añade al fichero Polinomio.h el prototipo de la siguiente función, y añade su implementación al fichero Polinomio.c. En el fichero Principal.c quita los comentarios al código que se proporciona para que puedas probar esta función.

```
/* Lee los coeficientes de un polinomio que están almacenados en
un fichero de binario, y crea la lista de monomios p. Los coeficientes en
el fichero de binario están en orden inverso, es decir, de menor exponente
a mayor exponente para facilitar la creación del polinomio. Los exponentes
no aparecen en el fichero. Por ejemplo, para el polinomio ejemplo el
fichero de binario estaría compuesto por la secuencia de enteros en binario
“95030203”.*/
int crearDeFichero(TPolinomio *p,char *nombre);
```

Ejercicio Polinomios-Tercera Parte

Implementa la siguiente función en el fichero Principal.c y quita los comentarios al código que se proporciona en el fichero Principal.c para probar esta función.

```
/* Dados dos polinomios p1 y p2, devuelve 1 si todos los monomios del
polinomio p1 son también monomios del polinomio p2, y 0 en caso contrario.
Implementar este algoritmo utilizando solo funciones de “Polinomio.h” */
int estaIncluido(TPolinomio p1,TPolinomio p2);
```

Añade al fichero Polinomio.h el prototipo de la siguiente función, y añade su implementación al fichero Polinomio.c. En el fichero Principal.c quita los comentarios al código que se proporciona para que puedas probar esta función.

```
/* Evalúa el polinomio para el valor x y devuelve el resultado. Para la
evaluación del polinomio se debe utilizar el método de Horner, de manera
que ax^4 + bx^3 + cx^2 + dx + e puede evaluarse en un valor cualquiera x teniendo
en cuenta que es equivalente a ((((ax + b)x + c)x + d)x + e. */
int evaluar(TPolinomio p,int x);
```

Anexo. Los prototipos de las funciones de lectura y escritura en ficheros de la biblioteca `<stdio.h>` son los siguientes (se dan por conocidos los prototipos de las funciones de `<stdlib.h>` que necesites, como `free` o `malloc`):

`FILE *fopen(const char *path, const char *mode);`
Abre el fichero especificado en el modo indicado ("rb"/ y "wb" para lectura/escritura binaria y "r"/"w" para lectura/escritura de texto). Devuelve un puntero al manejador del fichero en caso de éxito y NULL en caso de error.

`int fclose(FILE *fp);`
Guarda el contenido del buffer y cierra el fichero especificado. Devuelve 0 en caso de éxito y -1 en caso de error.

`unsigned fread(void *ptr, unsigned size, unsigned nmemb, FILE *stream);`
Lee `nmemb` elementos de datos, cada uno de tamaño `size bytes`, desde el fichero `stream`, y los almacena en la dirección apuntada por `ptr`. Devuelve el número de elementos leídos.

`int fscanf(FILE *stream, const char *format, ...)`

Lee del fichero `stream` los datos con el formato especificado en el parámetro `format` , el resto de parámetros son las variables en las que se almacenan los datos leídos en el formato correspondiente. La función devuelve el número de variables que se han leído con éxito.