Tecnólogo en Informática
EXAMEN Principios de Programación
29 de Julio de 2013

El examen suma 100 puntos.
Para aprobar se necesitan 60 puntos.

Ejercicio 1 (40 puntos)

a) Dado un arreglo de enteros, escriba una función de nombre *arregloOrdenado* que retorne true si el arreglo está ordenado de menor a mayor y false en caso contrario.

```
bool arregloOrdenado(int arr[], int n)
```

Donde *arr* es el arreglo de enteros y *n* es el tamaño del mismo.

Solución:

```
bool arregloOrdenado(int arr[], int n){

    bool ordenado = true;
    int i = 0;

    while ( (i < n-1) && (arr[i] < arr[i+1]) ){
            i++;
    }

    return (i == n-1);
}
```

b) Dados dos conjuntos de enteros representados con arreglos ordenados de menor a mayor, escriba una función de nombre *interseccionArreglos* que calcule la intersección de los conjuntos. El resultado debe ser un arreglo ordenado. Previo a calcular la intersección, la función debe chequear que los arreglos parámetro estén ordenados invocando a la función *arregloOrdenado* de la parte a). Si alguno de ellos no está ordenado, no realizará la intersección y desplegará en pantalla un mensaje de error.

```
void interseccionArreglos(int arr1[], int n,
int arr2[], int m, int interseccion[]);
```

Donde *arr1* y *arr2* son los arreglos de entrada y *n* y *m* son sus tamaños respectivamente. Asuma que el parámetro de salida, arreglo *interseccion* se invoca con un arreglo de tamaño *min(n,m)*.

Solución:

```
void interseccionArreglos(int arr1[], int n, int arr2[], int m, int interseccion[]){

    if (!arregloOrdenado(arr1,n))
      printf("Error: no se realiza la interseccion, arreglo 1 no ordenado\n");
    else if (!arregloOrdenado(arr2,m))
      printf("Error: no se realiza la interseccion, arreglo 2 no ordenado\n");
    else {

     int i = 0; //para desplazarse en arr1
     int j = 0; //para desplazarse en arr2
     ink k = 0; //para desplazarse en interseccion


     while (i < n && j < m){
          if (arr1[i]==arr1[j]){
            interseccion[k] = arr1[i];
            i++;
            j++;
            k++;
          }
          else if (arr1[i]<arr1[j])
                i++;
          else
                j++;
     }//while
    }//else
}
```

Ejercicio 2 (30 puntos)

Dada la siguiente representación de un polinomio:

```
#define tamanio 100

struct poli{
        int coef[tamanio];
        int grado;};
```

Ejemplos de representación:

Polinomio	Arreglo
$3+4x+2x^2$	[3|4|2]
$3x+4x^2+2x^3$	[0|3|4|2]
$5\ x^4$	[0|0|0|0|5]
7	[7]
7x	[0|7]
7+7x	[7|7]

Se pide: Defina una función que calcule el producto de dos polinomios.

Suponga dada una función que calcula la suma de dos polinomios con el siguiente prototipo:

```
poli  suma(poli p1,poli p2)
```

Recordando que dados los polinomios:

$a_0 + a_{1.}x + a_{2.}x^2 + \ldots\ldots\ldots + a_{n.}x^n$ y $b_0 + b_{1.}x + b_{2.}x^2 + \ldots\ldots\ldots + b_{m.}x^m$

La multiplicación se puede calcular como:

$a_{0.}(b_0 + b_{1.}x + b_{2.}x^2 + \ldots\ldots\ldots + b_{m.}x^{m)} + a_{1.(}b_0 .x + b_{1.}x^2 + \ldots\ldots\ldots + b_{m.}x^{m+1)} + a_{2.(}b_{0..}x^2 + \ldots\ldots\ldots + b_{m.}x^{m+2)} + \ldots\ldots\ldots + a_{n.(}\ b_{0.}x^n + b_{1.}x^{n+1} + \ldots\ldots\ldots + b_{m.}x^{n+m)}$

Se sugiere implementar las siguientes dos funciones auxiliares:

- La primera que dado un polinomio *p* y un escalar *e*, multiplica a *p* por *e* y devuelve el polinomio resultado.
- La segunda que dado un polinomio *p* y un entero *s*, corre los coeficientes de *p, s* posiciones hacia la derecha, agregando ceros en los primeros *s* lugares y devuelve el polinomio resultado.
  Es decir que por ejemplo dado el polinomio $3+4x+2x^2$ y s=1, devuelve el polinomio $3x+4x^2+2x^3$

Solución:

```
poli mult_poli_escalar(poli p, int e){
    poli res;

    for(int i=0; i < p.grado; i++){
          res.coef[i] = p.coef[i]*p;
    }

    res.grado = p.grado;
    return res;
}

poli shift_poli(poli p, int s){

    poli res;

    if (s <= 0){
      printf ("Error: el valor a desplazar debe ser positivo");
    }
    else {
        res.grado = p.grado + s;

        for(int i = 0; i < res.grado; i++){
          if (i < s)
              res.coef[i] = 0;
          else res.coef[i] = p.coef[i-s];
        }

    return res;
}

poli prod_poli(poli p1, poli p2){

    poli res, sw, term;

    //inicializo polinomio resultado
    res.grado = 0;
    res.coef[0] = 0;

    res.grado = p1.grado + p2.grado;

    for (int i = 0; i <= p1.grado; i ++){

        sw = poli shift_poli(p2,i);

        term = mult_poli_escalar(sw,p1.coef[i]);

        res = suma(res, term);
    }

    return res;
}
```

Ejercicio 3 (30 puntos)

Una empresa de ventas por correo vende cinco productos diferentes cuyos precios de lista son los siguientes:

Número de producto	precio
1	**3 $**
2	**4,5 $**
3	**5 $**
4	**6,5$**
5	**7 $**

Escriba un programa que lea una serie de pares de números con la siguiente información:

a) Número de producto
b) Cantidad vendida

hasta que se ingrese un número de producto igual a -1. Los números representan la venta de un día.

El programa debe calcular el importe total vendido en un día.

Solución:

```
main(){
    int num_prod = 0;

    int cant_vend = 0;

    float tot_vend = 0;

    do {

        printf("Ingrese el numero de producto:");
        scanf("%d", &num_prod);

        if (num_prod != -1){
           printf("Ingrese la cantidad vendida:");
           scanf("%d", &cant_vend);

           switch(num_prod){
              case 1:
                 tot_vend += 3*cant_vend;
                 break;
              case 2:
                 tot_vend +=4.5*cant_vend;
                 break;
              case 3:
                 tot_vend += 5*cant_vend;
                 break;
              case 4:
                 tot_vend += 6.5*cant_vend;
```

```
                    break;
                case 5:
                    tot_vend += 7*cant_vend;
                    break;
                default:
                      printf("No se ingreso un numero de producto valido\n");
            }
        }

    }while (num_prod != -1);

    printf("El importe total vendido en el dia es %f\n",tot_vend);


}
```