Estructuras de Datos y Algoritmos 1

Solución - Práctico 1

Tema: Introducción al Lenguaje C/C++

Ejercicios

2) Diseñar una rutina que, dadas dos variables, numerador y denominador simplifique la fracción numerador / denominador (sugerencia: usar el algoritmo de Euclides). Por ejemplo, 30/12 = 5/2

```
//*********************************************
//Resuelve el Ejercicio 2 del Practico 1

void ejercicio2(){
      int numerador = 0, denominador = 0, mcd = 0;

      numerador = obtenerNro("NUMERADOR: ");
      denominador = obtenerNro("DENOMINADOR: ");
      cout<<"La fraccion ingresada es: "<<numerador<<"/"<<denominador<<"\n";
      mcd = mcdFunc (numerador, denominador);
      mostrar(numerador, denominador, mcd);
}
//*********************************************
// Despliega el parámetro mensaje y retorna un entero leído por stdin
int obtenerNro(char* mensaje){
      int n;

      cout<<mensaje;
      cin>>n;
      return n;
}
//*********************************************
//Retorna el mcd entre dos enteros recibidos como parámetro
int mcdFunc (int a, int b){
      int resto, c;
      if (a < 0) a = -a;
      if (b < 0) b = -b;
      if (a < b){c = a; a = b; b = c;}
      assert (b != 0);
      do{
            resto = a%b;
            a = b;
            b = resto;
      }while(b != 0);
      return a;
}
```

```
//*********************************************
//Despliega la fraccion simplificada
void  mostrar(int num, int den, int mcd){
      cout<<"la fraccion simplificada: ";
      cout<<(long)num/mcd<<"/"<<(long)den/mcd<<"\n\n";
}
```

5)

a) **Leer n números (n será ingresado inicialmente por pantalla) e indicar su promedio, la cantidad de números pares, cantidad de múltiplos de 3.**

b) **Idem pero para una serie de números terminada por 0. ¿Qué hubo que cambiar?**

```
//*********************************************
//
 void ejercicio5a(){
 int n, nro, i, cantMult2, cantMult3, suma;

 n = obtenerNro("Ingresar la cantidad de numeros a leer: ");

 for (i = 0, suma = 0, cantMult2 = 0, cantMult3 = 0; i < n; i++){
      nro = obtenerNro("Ingresar nunero: ");
      if (esMult2(nro)) cantMult2 ++;
      if (esMult3(nro)) cantMult3 ++;
      suma += nro;
  }
 if (n > 0){
      cout<<"PROMEDIO: "<< (float) suma/n<<"\n";
      cout<<"CANTIDAD DE MULTIPLOS DE 2: "<< cantMult2 << "\n";
      cout<<"CANTIDAD DE MULTIPLOS DE 3: "<< cantMult3 << "\n";
 }
}

void ejercicio5b(){
 int cant, nro, cantMult2, cantMult3, suma;

 nro = obtenerNro("Ingresar numero: ");
 for (cant = 0, suma = 0, cantMult2 = 0, cantMult3 = 0; nro != 0; cant++){
      if (esMult2(nro)) cantMult2 ++;
      if (esMult3(nro)) cantMult3 ++;
      suma += nro;
      nro = obtenerNro("Ingresar nunero: ");
  }
 if (cant > 0){
  cout<<"PROMEDIO: "<< (float) suma/cant<<"\n";
  cout<<"CANTIDAD DE MULTIPLOS DE 2: "<< cantMult2 << "\n";
  cout<<"CANTIDAD DE MULTIPLOS DE 3: "<< cantMult3 << "\n";
 }

}
```

```
//*********************************************
//Dado un entero recibido como parámetro retorna true si es multimplo de dos, falso
en otro caso
int esMult2(int n){
      return (n % 2 == 0);
}

//*********************************************
//Dado un entero recibido como parámetro retorna true si es multimplo de tres,
falso en otro caso
int esMult3(int n){
      return (n % 3 == 0);
}
```

Aritmética de Direcciones

10) Hacer una función que intercambie el contenido de dos variables enteras recibidas como parámetro.

```
//*********************************************
//Resuelve el Ejercicio 10 del Practico 1

void intercambio(int *, int *);

void ejercicio10(){
 int nro1, nro2;

 nro1 = obtenerNro("Ingrese el primer nro: ");
 nro2 = obtenerNro("Ingrese el segundo nro: ");
 intercambio(&nro1, &nro2);
 cout<<"El primer número almacenado es: "<<nro1<<"\n";
 cout<<"El segundo número almacenado es: "<<nro2<<"\n\n";
}

void intercambio(int *n1, int *n2){
      int aux;

      aux = *n1;
      *n1 = *n2;
      *n2 = aux;
}
```

11) Indique que hace la siguiente función:

```
int misterio (char *p){
      char *t = p;
      while (*t != '\0' )
            t++;
      return t – p;
}
```

```
//*********************************************
//Resuelve el Ejercicio 11 del Practico 1
int misterio11 (char *);

void ejercicio11(){
      char *linea = "Este es un string de prueba del Ejercicio 11";
      cout<<"La entrada de la funcion es: " << linea<<"\n\n";
      cout<<"La salida de la funcion es: " << misterio11(linea)<<"\n";
      cout<<"El largo de la entrada\n\n";
}

int misterio11 (char *p){ //strlen
      char *t = p;
      while (*t != '\0' )
            t++;
      return t - p;
}
```

12) Indique que hace la siguiente función:

```
void misterio (char *p, char *t){

      while (*p++ = *t++);
}

//*********************************************
//Resuelve el Ejercicio 12 del Practico 1

void misterio12 (char *p, char *t);

void ejercicio12(){
      char *lineaO = "Este es un string de prueba del Ejercicio 12";
      char *lineaD = (char*) malloc (strlen(lineaO)+1);

      cout<<"La entrada de la funcion es: " << lineaO<<"\n\n";
      misterio12(lineaD, lineaO);
      cout<<"La salida de la funcion es: " << lineaD<<"\n";
      cout<<"Copia el segundo en el primero\n\n";
}
```

Arreglos y matrices

16)

a) Escriba una función int maximoDelVector (vector v) que devuelve el máximo encontrado dentro del vector v.

b) Escriba una función int posicionDelMaximoDelVector (vector v) que devuelve la posición del máximo encontrado dentro del vector v.

```
//*********************************************
//Resuelve el Ejercicio 16 del Practico 1

int maximoDelVector(int [], int);

void ejercicio16a(){
      int v[] = {10, 2, 3, 5, 13, 11, 20, 19, 15, 0};
      int max = maximoDelVector(v, N);
      muestroVec(v, N);
      cout<<"El maximo es: " << max << "\n\n";
}

int maximoDelVector(int v[], int t)
{     int i, maximo = v[0];
      for (i = 1; i < t; i++)
            if (v[i] > maximo)
                  maximo = v[i];
      return maximo;
}

int posicionDelMaximoDelVector(int [], int);

void ejercicio16b(){
      int v[] = {10, 2, 3, 5, 13, 11, 20, 19, 15, 0};
      int pos = posicionDelMaximoDelVector(v, N);
      muestroVec(v, N);
      cout<<"La posicion es: " << pos << "\n\n";

}

int posicionDelMaximoDelVector(int v[], int t){
      int i, posicion = 0;
      for (i = 0; i < t; i++)
            if (v[i] > v[posicion])
                  posicion = i;
      return posicion;

}

int posicionDelMinimoDelVector(int v[], int t){
      int i, posicion = 0;
      for (i = 0; i < t; i++)
            if (v[i] > v[posicion])
                  posicion = i;
      return posicion;

}
```

17) Escriba una rutina que reciba un vector y lo ordene.

```
//Resuelve el Ejercicio 17 del Practico 1
void ordenacion(int *,int);

void ejercicio17(){
	char a;
	int v[] = {10, 2, 3, 5, 13, 11, 20, 19, 15, 0};
	muestroVec(v, N);
	cin>>a;
	ordenacion(v,N);
	muestroVec(v, N);
}


void ordenacion(int vec[], int tope){
	int posMax, t = tope - 1;

	for (;t >= 0; t--){
		posMax = posicionDelMaximoDelVector(vec, t + 1);
		intercambio(&vec[t], &vec[posMax]);
	}
}
```

19) Dados dos vectores ordenados v1 y v2 de dimensiones n1 y n2 respectivamente escribir una función que genere un tercer vector v3 que contenga los elementos de los dos anteriores intercalados manteniendo el orden.

```
//Resuelve el Ejercicio 19 del Practico 1
void apareo(int [], int [], int []);

void ejercicio19(){
	int vec1[] = { 0, 2, 4, 6, 8, 10, 12, 14, 16, 18};
	int vec2[] = { 1, 3, 5, 7, 9, 11, 13, 15, 17, 19};
	int vecRes[2*N];

	muestroVec(vec1, N);
	muestroVec(vec2, N);
	apareo(vec1, vec2, vecRes);
	muestroVec(vecRes, 2*N);
}
```

```
void apareo(int v1[], int v2[], int vr[]){
      int i = 0, j = 0, t = 0;

      while (i < N && j < N){
            if (v1[i] == v2[j]){
                  vr[t] = v1[i++];
                  vr[t++] = v2[j++];
            }
            else{
                  if (v1[i] < v2[j]){
                        vr[t++] = v1[i++];
                  }
                  else{
                        vr[t++] = v2[j++];
                  }
            }

      }
      while (i < N) vr[t++] = v1[i++];
      while (j < N) vr[t++] = v2[j++];
}
```