|  |  |
| --- | --- |
|  |  |
| Test Name/# | 1 |
| **Test Objective** | Encontrar las loc´s por función |
| **Test Description** | Introducir el código del programa 1A |
| **Test Conditions** | /\*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++  Version del programa: 1.3  Nombre del ingeniero: Juan Alberto Gutierrez Canto  Fecha de creación: 28/01/2016  Descripcion: Programa 1A de la materia proceso personal de software, programa que calcula  la media y desviacion estandar de N numeros con listas ligadas  +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++  \*/  #include <cstdlib>  #include <stdio.h>  #include <malloc.h>  #include <math.h>  #include <iostream>  /\*+++++++++++++Reutilizacion +++++++++++++++++++  Function: ISEMPTY()  Propósito: revisar si la lista esta vacia  In/out: no aplica/1 esta vacia-0 no esta vacia  Limitaciones: no aplica  +++++++++++++++++++++++++++++++++++++++++++  \*/  using namespace std;  struct nodo  { /\*-c estructura para guardar las listas ligadas\*/  double numero;  struct nodo \*sig;  };struct nodo \*raiz=NULL, \*last=NULL;  int ISEMPTY()  { /\*-c funcion para saber si esta vacia las listas ligadas\*/  if(raiz == NULL) /\*-c esta vacia la lista?? 1-si 0-no\*/  return 1;  else  return 0;  }  /\*+++++++++++++Reutilizacion +++++++++++++++++++  Function: INSERT()  Propósito: insertar un dato en un nuevo nodo  In/out: dato a insertar/no aplica  Limitaciones: tipo de dato double  +++++++++++++++++++++++++++++++++++++++++++  \*/  void INSERT(double dato)  { /\*-c insertar un dato en un nuevo nodo\*/  struct nodo \*nuevo = NULL; /\*-c nuevo nodo\*/  if(ISEMPTY())  {  nuevo = (struct nodo \*)malloc( sizeof (struct nodo)); /\*-c inserta si no hay ninguna lista todavia\*/  nuevo->numero = dato;  nuevo-> sig = NULL;  raiz = last = nuevo;  }  else  {  nuevo = (struct nodo \*)malloc( sizeof (struct nodo)); /\*-c insertar cuando ya existe un nodo\*/  nuevo->numero = dato;  nuevo-> sig = NULL;  last->sig = nuevo;  last = nuevo;  }  }  /\*+++++++++++++Reutilizacion +++++++++++++++++++  Function: REMUEVE()  Propósito: remueve un dato de la lista elimina el nodo  In/out: no aplica/dato  Limitaciones: tipo de dato double  +++++++++++++++++++++++++++++++++++++++++++  \*/  double REMUEVE()  { /\*-c regresar dato de la lista ligada\*/  struct nodo \*elimina = NULL;  double temp;  if(!ISEMPTY())  {  elimina = raiz;  raiz = raiz-> sig; /\*-c o raiz = elimina -> sig;\*/  temp = elimina -> numero;  free(elimina); /\*-c libera espacio de memoria\*/  return temp;  }  else  return '\0';  }  int main()  {  double media=0, desviacion=0,numero;  int n;  cin>>n;  for(int i=0;i<n;i++) /\*-c lee el ingreso de los datos de 0 a n\*/  {  cin>>numero;  INSERT(numero); /\*-c guardamos en la lista ligadas\*/  media+=numero; /\*-c calculamos la sumatoria para la media\*/  }  media=media/n; /\*-c calculamos la media\*/  for(int i=0;i<n;i++) /\*-c ciclo para hacer sumatoria de desviacion estandar\*/  {  desviacion+=pow((REMUEVE()-media),2);  }  desviacion= pow((desviacion/(n-1)),.5); /\*-c terminamos de calcular la desviacion estandar\*/  cout<<"Media "<<media<<endl<<"Desviacion "<<desviacion<< endl; /\*-c imprime resultados\*/  system("PAUSE");  return 0;  } |
| **Expected Results** | Función :  Total: |
| **Actual Results** |  |

|  |  |
| --- | --- |
| Test Name/# | 2 |
| **Test Objective** | Encontrar las loc´s por función |
| **Test Description** | Introducir el código del programa 2A |
| **Test Conditions** | /\*+++++++++++++++++++++++Program++++++++++++++++++++++++  Version: 1.3  Nombre: Juan Alberto Gutierrez Canto  Fecha: 11/02/2016  Descripcion: Programa 2A de la materia proceso personal de software, programa que cuenta las loc´s del codigo  +++++++++++++++++++++++++++++++++++++++++++++++++++++++  \*/  /\*+++++++++++++++++++++Contenido++++++++++++++++++++++++  LOC reusadas: 0  LOC modificadas: 15  LOC compilación : 146  Librerias:  <iostream>  <stdio.h>  <string>  Clases: N/A  Funciones:  void FUNCION();  Surce code in C:\Users\equipo\Documents\Personal Process Software\2A\main.cpp  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++  \*/  #include <iostream>  #include <stdio.h>  #include <string>  using namespace std;  int contador\_comentario,contador\_globallocs=0;  /\*++++++++++++++++++++ seccion ++++++++++++++++++++++++++++++++++++++++++++++++++++++++  Function: FUNCION()  Propósito: cuanta las loc's logicas y los comentarios  In/out: direccion del archivo/no aplica  Limitaciones: ninguna  +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++  \*/  void FUNCION(FILE \*archivo\_lectura)  {  char caracter[200];  int contador\_funcion=0,i;  fgets(caracter,200,archivo\_lectura);  if(caracter[0] == '{')  {  while(caracter[0] != '}')  {  fgets(caracter,200,archivo\_lectura);  i=200;  while(i--)  {  if(caracter[i] == '/' && caracter[i-1] == '\*')  {  caracter[i]='2';  contador\_comentario++;  break;  }  }  contador\_funcion++;  }  }  contador\_funcion++;  cout<<contador\_funcion<<" Locs\n\n";  contador\_globallocs+=contador\_funcion;  }  int main()  {  int numero\_archivo=1;  char archivo[30],caracter[205];  FILE \*archivo\_lectura;  while(numero\_archivo){  cout<<"Primero copia el archivo que desea contar \na la carpeta del programa \n";  cout<<"una vez echo esto:\n\nIntroduce el nombre del codigo fuente a analizar"<<endl;  cin>>archivo;  archivo\_lectura=fopen(archivo,"r");/\*-c se habre el codigo a revisar\*/  contador\_comentario=0;/\*-I la variable es contador de loc comentario\*/  contador\_globallocs=0;/\*-Ila variable es contador de loc general\*/  if(archivo\_lectura == NULL)/\*-c verifica si existe el archivo\*/  cout<<"No se encontro el archivo";  else  {  while (feof(archivo\_lectura) == 0)/\*-c recorrer el archivo hasta que se acabe\*/  {  fgets(caracter,200,archivo\_lectura);/\*-c funcion para cachar lineas del archivo\*/  if(caracter[0] == '/' && caracter[1] == '\*')/\*-c encontrar comentarios de caja\*/  {  while(caracter[0] != '\*' && caracter[1] != '/')/\*-c saltar lineas de comentarios de caja\*/  {  fgets(caracter,200,archivo\_lectura);  }  contador\_comentario++;  }  if(caracter[0] == 'i' && caracter[1] == 'n' && caracter[2] == 't')/\*-c cuenta loc's logicas y comentario funcion int\*/  {  cout<<"Funcion .- "<<caracter;  FUNCION(archivo\_lectura);  fgets(caracter,200,archivo\_lectura);  }  if(caracter[0] == 'v' && caracter[1] == 'o' && caracter[2] == 'i')/\*-c cuenta loc's logicas y comentario funcion void\*/  {  cout<<"Funcion .- "<<caracter;  FUNCION(archivo\_lectura);  }  if(caracter[0] == 'f' && caracter[1] == 'l' && caracter[2] == '0')/\*-c cuenta loc's logicas y comentario funcion float\*/  {  cout<<"Funcion .- "<<caracter;  FUNCION(archivo\_lectura);  }  if(caracter[0] == 'c' && caracter[1] == 'h' && caracter[2] == 'a')/\*-c cuenta loc's logicas y comentario funcion char\*/  {  cout<<"Funcion .- "<<caracter;  FUNCION(archivo\_lectura);  }  if(caracter[0] == 's' && caracter[1] == 't' && caracter[2] == 'r')/\*-c cuenta loc's logicas y comentario funcion string\*/  {  cout<<"Funcion .- "<<caracter;  FUNCION(archivo\_lectura);  }  if(caracter[0] == 'd' && caracter[1] == 'o' && caracter[2] == 'u')/\*-c cuenta loc's logicas y comentario funcion double\*/  {  cout<<"Funcion .- "<<caracter;  FUNCION(archivo\_lectura);  }  if(caracter[0] == 'c' && caracter[1] == 'l' && caracter[2] == 'a')/\*-c cuenta loc's logicas y comentario clases\*/  {  cout<<"Clase .- "<<caracter;  FUNCION(archivo\_lectura);  }  if(caracter[0] == '#')/\*-c cuenta loc's define y librerias\*/  {  contador\_globallocs++;  }  }  contador\_globallocs+=contador\_comentario;  cout<<"Comentarios "<<contador\_comentario<<" Loc's\n\n";  cout<<"Lineas totales de codigo(loc's) --"<<contador\_globallocs<<"--\n";  }  cout<<"\nDesea leer otro archivo?\n1=si 0=no\n";  cin>>numero\_archivo;  }  return 0;  } |
| **Expected Results** | Función :  Total: |
| **Actual Results** |  |

|  |  |
| --- | --- |
| Test Name/# | 3 |
| **Test Objective** | Encontrar las loc´s por función |
| **Test Description** | Introducir el código del programa 3A |
| **Test Conditions** | /\*+++++++++++++++++++++++Program++++++++++++++++++++++++  Version: 1.3  Nombre: Juan Alberto Gutierrez Canto  Fecha: 21/02/2016  Descripcion: Programa 3A de la materia proceso personal de software, programa que cuenta las loc´s del codigo por funciones  +++++++++++++++++++++++++++++++++++++++++++++++++++++++  \*/  /\*+++++++++++++++++++++Contenido++++++++++++++++++++++++  LOC reusadas: 28  LOC modificadas: 30  LOC compilación : 244  Librerias:  <iostream>  <stdio.h>  <string>  Clases: N/A  Funciones:  void FUNCION();  Surce code in C:\Users\equipo\Documents\Personal Process Software\PROGRAMA 3A JUAN ALBERTO GUTIERREZ CANTO\3-Codificacion\main.cpp  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++  \*/  #include <iostream>  #include <stdio.h>  #include <string>  using namespace std;  int contador\_comentario,contador\_globallocs=0,contador\_globales,contador\_defineinclude;  /\*++++++++++++++++++++ Reutilizacion ++++++++++++++++++++++++++++++++++++++++++++++++++++++++  Function: FUNCION()  Propósito: cuenta las loc's logicas y los comentarios  In/out: direccion del archivo/no aplica  Limitaciones: ninguna  +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++  \*/  void FUNCION(FILE \*archivo\_lectura)  {  char caracter[200];  int contador\_funcion=0,i;  fgets(caracter,200,archivo\_lectura);  if(caracter[0] == '{')  {  while(caracter[0] != '}')  {  fgets(caracter,200,archivo\_lectura);  i=200;  while(i--)  {  if(caracter[i] == '/' && caracter[i-1] == '\*')  {  caracter[i]='2';  contador\_comentario++;  break;  }  }  contador\_funcion++;  }  contador\_funcion++;  cout<<contador\_funcion<<" Locs\n\n";  }  contador\_globallocs+=contador\_funcion;  }  int main()  {  int numero\_archivo=1,chek=0;  char archivo[30],caracter[205];  FILE \*archivo\_lectura;  while(numero\_archivo){  cout<<"Primero copia el archivo que desea contar \na la carpeta del programa \n";  cout<<"una vez echo esto:\n\nIntroduce el nombre del codigo fuente a analizar"<<endl;  cin>>archivo;  archivo\_lectura=fopen(archivo,"r");/\*-c se abre el codigo a revisar\*/  contador\_comentario=0;/\*-I la variable es contador de loc comentario\*/  contador\_globallocs=0;/\*-I la variable es contador de loc general\*/  contador\_globales=0;/\*-I la variable es contador de loc de incializacion de variables globales\*/  contador\_defineinclude=0;/\*-I la variable es contador de loc de define y includes\*/  if(archivo\_lectura == NULL)/\*-c verifica si existe el archivo\*/  cout<<"No se encontro el archivo";  else  {  while (feof(archivo\_lectura) == 0)/\*-c recorrer el archivo hasta que se acabe\*/  {  fgets(caracter,200,archivo\_lectura);/\*-c funcion para cachar lineas del archivo\*/  if(caracter[0] == '/' && caracter[1] == '\*')/\*-c encontrar comentarios de caja\*/  {  while(caracter[0] != '\*' && caracter[1] != '/')/\*-c saltar lineas de comentarios de caja\*/  {  fgets(caracter,200,archivo\_lectura);  }  contador\_comentario++;  }  if(caracter[0] == 'i' && caracter[1] == 'n' && caracter[2] == 't')/\*-c cuenta loc's logicas y comentario funcion int\*/  {  for(int v=200;v>0;v--)  {  if(caracter[v] == ';')  {  chek=1;  }  }  if(chek!=1)  {  cout<<"Funcion .- "<<caracter;  FUNCION(archivo\_lectura);  fgets(caracter,200,archivo\_lectura);  }  else  {  contador\_globales++;  }  chek=0;  }  if(caracter[0] == 'v' && caracter[1] == 'o' && caracter[2] == 'i')/\*-c cuenta loc's logicas y comentario funcion void\*/  {  for(int v=200;v>0;v--)  {  if(caracter[v] == ';')  {  chek=1;  }  }  if(chek!=1)  {  cout<<"Funcion .- "<<caracter;  FUNCION(archivo\_lectura);  fgets(caracter,200,archivo\_lectura);  }  else  {  contador\_globales++;  }  chek=0;  }  if(caracter[0] == 'f' && caracter[1] == 'l' && caracter[2] == '0')/\*-c cuenta loc's logicas y comentario funcion float\*/  {  for(int v=200;v>0;v--)  {  if(caracter[v] == ';')  {  chek=1;  }  }  if(chek!=1)  {  cout<<"Funcion .- "<<caracter;  FUNCION(archivo\_lectura);  fgets(caracter,200,archivo\_lectura);  }  else  {  contador\_globales++;  }  chek=0;  }  if(caracter[0] == 'c' && caracter[1] == 'h' && caracter[2] == 'a')/\*-c cuenta loc's logicas y comentario funcion char\*/  {  for(int v=200;v>0;v--)  {  if(caracter[v] == ';')  {  chek=1;  }  }  if(chek!=1)  {  cout<<"Funcion .- "<<caracter;  FUNCION(archivo\_lectura);  fgets(caracter,200,archivo\_lectura);  }  else  {  contador\_globales++;  }  chek=0;  }  if(caracter[0] == 's' && caracter[1] == 't' && caracter[2] == 'r')/\*-c cuenta loc's logicas y comentario funcion string\*/  {  for(int v=200;v>0;v--)  {  if(caracter[v] == ';')  {  chek=1;  }  }  if(chek!=1)  {  cout<<"Funcion .- "<<caracter;  FUNCION(archivo\_lectura);  fgets(caracter,200,archivo\_lectura);  }  else  {  contador\_globales++;  }  chek=0;  }  if(caracter[0] == 'd' && caracter[1] == 'o' && caracter[2] == 'u')/\*-c cuenta loc's logicas y comentario funcion double\*/  {  for(int v=200;v>0;v--)  {  if(caracter[v] == ';')  {  chek=1;  }  }  if(chek!=1)  {  cout<<"Funcion .- "<<caracter;  FUNCION(archivo\_lectura);  fgets(caracter,200,archivo\_lectura);  }  else  {  contador\_globales++;  }  chek=0;  }  if(caracter[0] == 'c' && caracter[1] == 'l' && caracter[2] == 'a')/\*-c cuenta loc's logicas y comentario clases\*/  {  cout<<"Clase .- "<<caracter;  FUNCION(archivo\_lectura);  }  if(caracter[0] == '#')/\*-c cuenta loc's define y librerias\*/  {  contador\_defineinclude++;  }  }  contador\_globallocs+=contador\_comentario;  contador\_globallocs+=contador\_globales;  contador\_globallocs+=contador\_defineinclude;  cout<<"Comentarios "<<contador\_comentario<<" Loc's\n\n";  cout<<"define & includes "<<contador\_defineinclude<<" Loc's\n\n";  cout<<"Variables globales "<<contador\_globales<<" Loc's\n\n";  cout<<"Lineas totales de codigo(loc's) --"<<contador\_globallocs<<"--\n";  }  cout<<"\nDesea leer otro archivo?\n1=si 0=no\n";  cin>>numero\_archivo;  }  return 0;  } |
| **Expected Results** | Función :  Total: |
| **Actual Results** |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Program Number | Object Name | Number of Methods | Object LOC | Total Program LOC |
| 1 A | struct nodo | 1 | 3 |  |
|  | void INSERT | 1 | 18 |  |
|  | double REMUEVE | 1 | 14 |  |
|  | int main | 1 | 20 |  |
|  | Comentarios | 19 | 19 |  |
|  | Define & includes | 5 | 5 |  |
|  | Variables globales | 2 | 2 |  |
|  |  |  |  | 78 |
| 2A | void FUNCION | 27 | 1 |  |
|  | int main | 77 | 1 |  |
|  | Comentarios | 19 | 19 |  |
|  | define &includes | 3 | 3 |  |
|  | variables globales | 1 | 1 |  |
|  |  |  |  | 127 |
| 3ª | void FUNCION | 26 | 1 |  |
|  | int main | 178 | 1 |  |
|  | Comentarios | 19 | 19 |  |
|  | define &includes | 3 | 3 |  |
|  | variables globales | 1 | 1 |  |
|  |  |  |  | 229 |