

Estructura de Datos y Análisis de Algoritmos Manual de usuario

**Andrés Felipe Muñoz Bravo
19.646.487-5**

Profesor:
Mario Inostroza Ponta
Ayudante:
Javiera Torres Muñoz

Santiago - Chile
2016

TABLA DE CONTENIDOS

Tabla de contenido

Tabla de Contenidos.....	3
Índice de Figuras	4
CAPÍTULO 1. Introducción	5
CAPÍTULO 2. Como compilar y ejecutar	6
2.1 Compilar y ejecutar en Windows:	6
2.2 Compilar y ejecutar en Linux:.....	8
CAPÍTULO 3. Funcionalidades del programa.....	10
3.1 Cargar grafo.	10
3.2 Mostrar características del grafo.	11
3.3 Verificar si el grafo es conexo.....	12
3.4 Ordenar por centralidad de grado.....	12
3.5 Ordenar por centralidad de betweenness.....	12
CAPÍTULO 4. Posibles errores.	13

ÍNDICE DE FIGURAS

Ilustración 1: Abrir consola de Windows.....	6
Ilustración 2: Buscar carpeta en consola de Windows.	7
Ilustración 3: Compilar en Windows.	7
Ilustración 4: Ejecutar programa en Windows.	7
Ilustración 5: Consola en Linux.	8
Ilustración 6: Buscar carpeta en Linux.....	8
Ilustración 7: Compilar en Linux.	9
Ilustración 8: Ejecutar programa en Linux.....	9
Ilustración 9: Funcionalidades	10
Ilustración 10: Formato archivo de texto.	10
Ilustración 11: Cargar grafo, éxito.	11
Ilustración 12: Cargar grafo, fallido.	11
Ilustración 13: Mostrar características del grafo.	11
Ilustración 14: Verificar grafo, conexo.	12
Ilustración 15: Verificar grafo, desconexo.	12
Ilustración 16: Centralidad de grado.	12

CAPÍTULO 1. INTRODUCCIÓN

Este programa es utilizado para analizar 3 características de grafos obtenidos desde un archivo texto. La primera característica, es verificar si el grafo es conexo, esto quiere decir que exista un camino entre todo par de vértice perteneciente al grafo, para esto se aplica un algoritmo de búsqueda dejando marcas por todos los vértices que han sido visitados y finalmente comprobar si se han visitado todos los nodos. La segunda característica, es ordenar los vértices según su centralidad de grado, en otras palabras, ordenar de mayor a menor, según la cantidad de vértices adyacentes a dicho vértice. La tercera característica, es ordenar los vértices según la centralidad de betweenness, lo que significa ordenar de mayor a menor, según el cociente entre la cantidad de veces que un nodo está presente entre los caminos mínimos entre dos nodos y el total de esos caminos mínimos.

CAPÍTULO 2. COMO COMPILAR Y EJECUTAR

Tenemos que tener en claro los siguientes comandos, que son útiles dentro de la consola de Windows y de Linux.

- “cd nombre_de_carpeta” para acceder a una carpeta que exista en la dirección actual
- “cd..” para volver una carpeta atras.
- “dir” para mostrar todos los archivos y carpetas existentes en la dirección actual.

2.1 COMPILAR Y EJECUTAR EN WINDOWS:

- 1) Abrir la consola: Presionar las teclas Windows+r e ingresar cmd, luego presionar en aceptar.

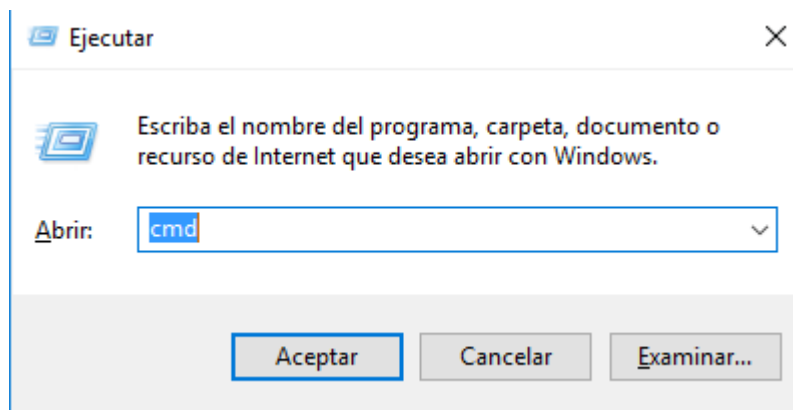
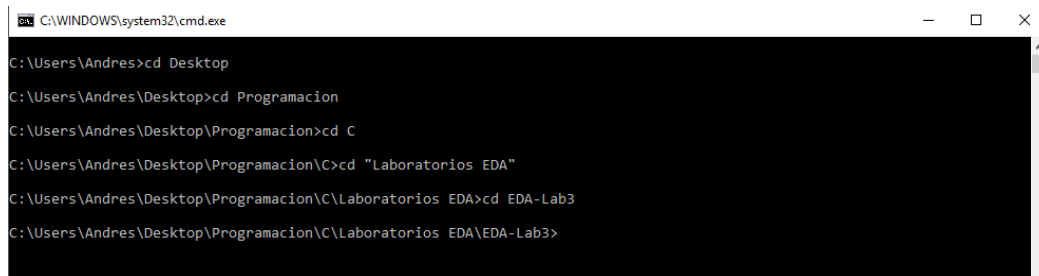


Ilustración 1: Abrir consola de Windows.

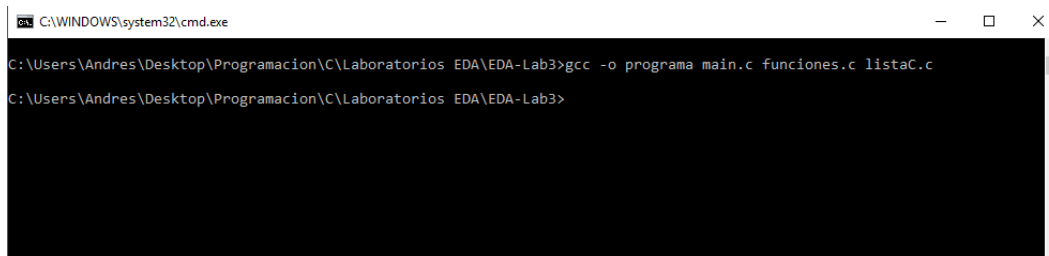
- 2) Buscar la carpeta contenedora de los archivos: Se utiliza el comando `cd` seguido de la ruta de la carpeta en la que se encuentran los archivos.



```
C:\WINDOWS\system32\cmd.exe
C:\Users\Andres>cd Desktop
C:\Users\Andres\Desktop>cd Programacion
C:\Users\Andres\Desktop\Programacion>cd C
C:\Users\Andres\Desktop\Programacion\C>cd "Laboratorios EDA"
C:\Users\Andres\Desktop\Programacion\C\Laboratorios EDA>cd EDA-Lab3
C:\Users\Andres\Desktop\Programacion\C\Laboratorios EDA\EDA-Lab3>
```

Ilustración 2: Buscar carpeta en consola de Windows.

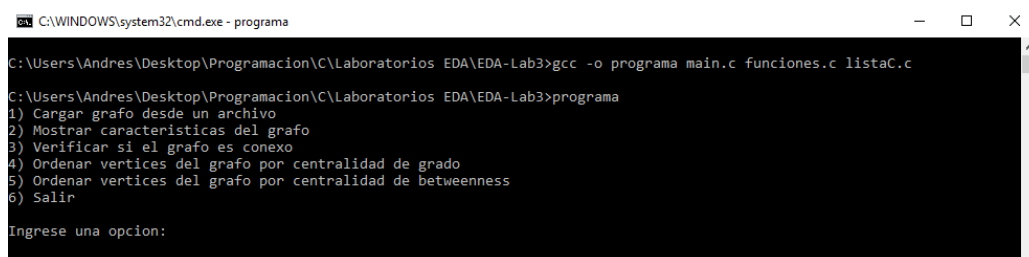
- 3) Compilar los archivos: Se utiliza el comando “`gcc -o programa main.c funciones.c listaC.c`”. Notar que en `listaC.c`, existe una “c” mayúscula.



```
C:\WINDOWS\system32\cmd.exe
C:\Users\Andres\Desktop\Programacion\C\Laboratorios EDA\EDA-Lab3>gcc -o programa main.c funciones.c listaC.c
C:\Users\Andres\Desktop\Programacion\C\Laboratorios EDA\EDA-Lab3>
```

Ilustración 3: Compilar en Windows.

- 4) Ejecutar el programa: Se ingresa en la consola “programa” o doble click en el ejecutable creado al momento de compilar.



```
C:\WINDOWS\system32\cmd.exe - programa
C:\Users\Andres\Desktop\Programacion\C\Laboratorios EDA\EDA-Lab3>gcc -o programa main.c funciones.c listaC.c
C:\Users\Andres\Desktop\Programacion\C\Laboratorios EDA\EDA-Lab3>programa
1) Cargar grafo desde un archivo
2) Mostrar características del grafo
3) Verificar si el grafo es conexo
4) Ordenar vertices del grafo por centralidad de grado
5) Ordenar vertices del grafo por centralidad de betweenness
6) Salir
Ingrese una opcion:
```

Ilustración 4: Ejecutar programa en Windows.

2.2 COMPILAR Y EJECUTAR EN LINUX:

Se siguen prácticamente los mismos pasos que en Windows.

- 1) Abrir la consola: Abrir la consola o terminal desde el inicio.

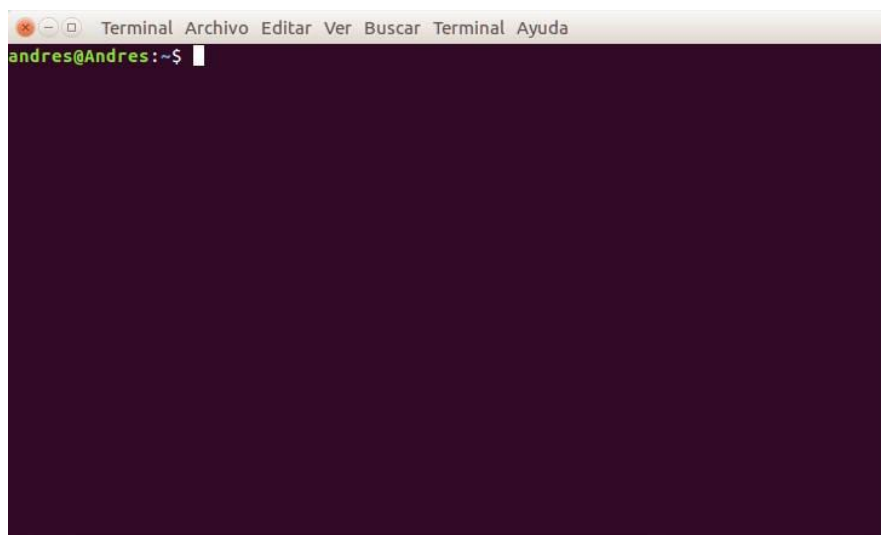


Ilustración 5: Consola en Linux.

- 2) Buscar la carpeta contenedora de los archivos: Se utiliza el comando `cd` seguido de la ruta de la carpeta en la que se encuentran los archivos.

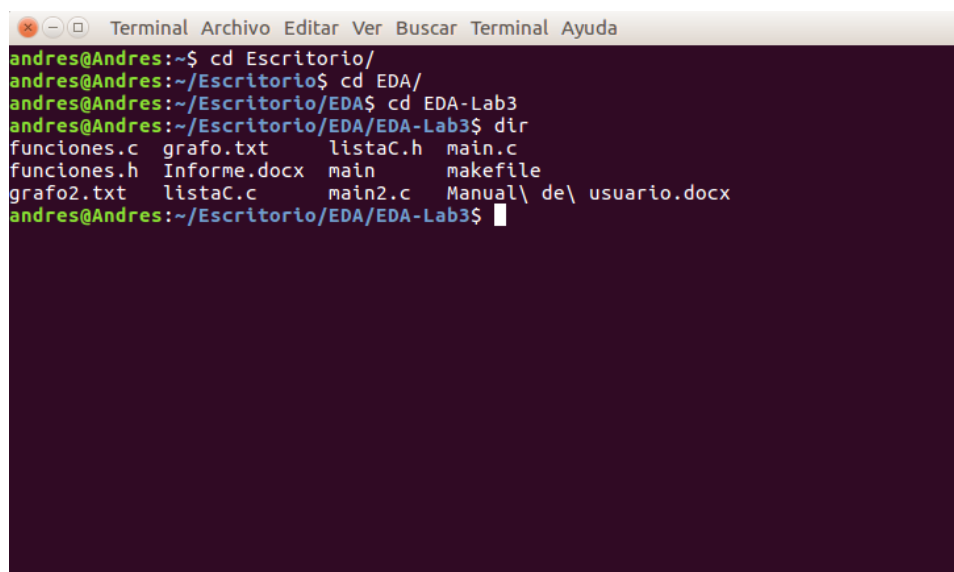
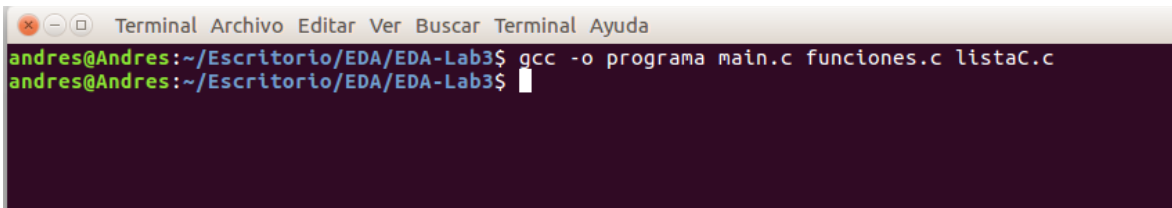


Ilustración 6: Buscar carpeta en Linux.

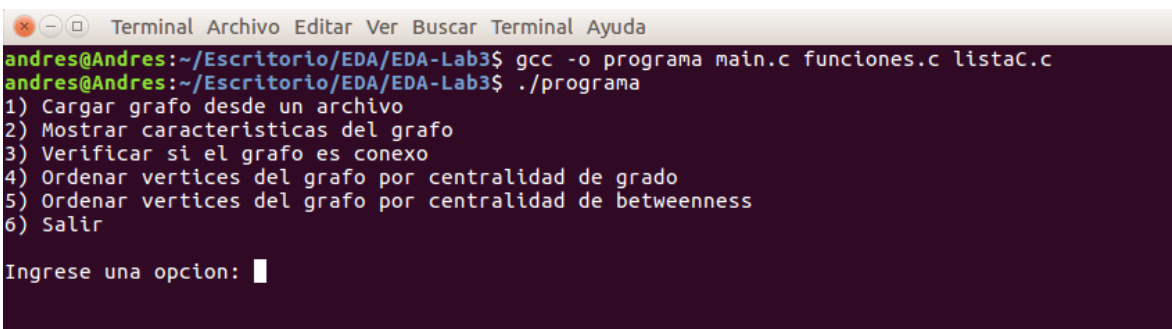
- 3) Compilar los archivos: Se utiliza el comando “gcc -o programa main.c funciones.c listaC.c”. Notar que listaC.c contiene una “c” mayúscula.



```
Terminal Archivo Editar Ver Buscar Terminal Ayuda
andres@Andres:~/Escritorio/EDA/EDA-Lab3$ gcc -o programa main.c funciones.c listaC.c
andres@Andres:~/Escritorio/EDA/EDA-Lab3$
```

Ilustración 7: Compilar en Linux.

- 4) Ejecutar el programa: Se ingresa en la consola “./programa”. Tener cuidado en este paso ya que no sirve solo poner “programa” como en el caso de Windows. Es necesario poner “./” antes del ejecutable.

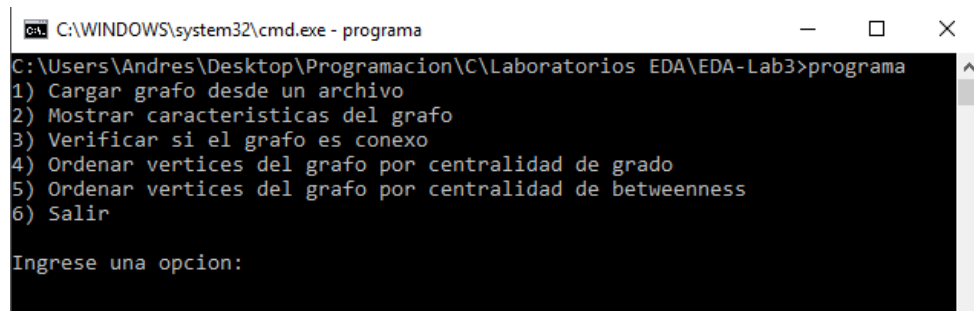


```
Terminal Archivo Editar Ver Buscar Terminal Ayuda
andres@Andres:~/Escritorio/EDA/EDA-Lab3$ gcc -o programa main.c funciones.c listaC.c
andres@Andres:~/Escritorio/EDA/EDA-Lab3$ ./programa
1) Cargar grafo desde un archivo
2) Mostrar características del grafo
3) Verificar si el grafo es conexo
4) Ordenar vertices del grafo por centralidad de grado
5) Ordenar vertices del grafo por centralidad de betweenness
6) Salir
Ingrese una opcion: 
```

Ilustración 8: Ejecutar programa en Linux.

CAPÍTULO 3. FUNCIONALIDADES DEL PROGRAMA.

Este programa posee cinco funcionalidades que se pueden elegir dentro del programa, estas se muestra a continuación:



```

C:\WINDOWS\system32\cmd.exe - programa
C:\Users\Andres\Desktop\Programacion\C\Laboratorios EDA\EDA-Lab3>programa
1) Cargar grafo desde un archivo
2) Mostrar características del grafo
3) Verificar si el grafo es conexo
4) Ordenar vertices del grafo por centralidad de grado
5) Ordenar vertices del grafo por centralidad de betweenness
6) Salir

Ingrese una opcion:
  
```

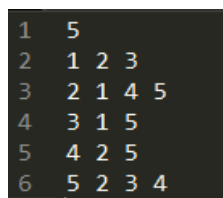
Ilustración 9: Funcionalidades

Para poder elegir alguna funcionalidad es necesario ingresar el número correspondiente a la funcionalidad deseada.

3.1 CARGAR GRAFO.

Con esta funcionalidad es posible cargar un grafo desde algún archivo de texto cabe mencionar de que el archivo de texto debe contener:

1. Primera línea: cantidad de vértices del grafo
2. Resto de las líneas: vértice con sus respectivos vértices adyacentes separados por un espacio.
3. Los vértices deben ser enumerados desde 1 hasta n.



```

1 5
2 1 2 3
3 2 1 4 5
4 3 1 5
5 4 2 5
6 5 2 3 4
  
```

Ilustración 10: Formato archivo de texto.

Una vez elegida la opción es necesario ingresar el nombre del archivo de texto donde se encuentra el grafo con lo anteriormente mencionado.

```

Ingrese una opcion: 1
Ingrese el nombre de un archivo: grafo5.txt
Grafo cargado

```

Ilustración 11: Cargar grafo, éxito.

```

Ingrese una opcion: 1
Ingrese el nombre de un archivo: grafo3.txt
El archivo no existe

```

Ilustración 12: Cargar grafo, fallido.

3.2 MOSTRAR CARACTERISTICAS DEL GRAFO.

Esta funcionalidad muestra por consola las siguientes características del grafo.

1. Vértices: Cantidad de vértices pertenecientes al grafo.
2. Lista de adyacencia: Lista que contiene todos los vértices adyacentes a cada nodo.
3. Matriz de adyacencia: Matriz que contiene un 1 en la posición donde dos vértices sean adyacentes.

Teniendo un grafo cargado previamente es posible llevar a cabo esta funcionalidad. A modo de ejemplo utilizaremos el grafo cargado en la funcionalidad anterior.

```

Ingrese una opcion: 2

GRAFO:
Vertices: 5
Lista adyacencia:
Vertice 1: 2 3
Vertice 2: 1 4 5
Vertice 3: 1 5
Vertice 4: 2 5
Vertice 5: 2 3 4
Matriz de adyacencia:
0 1 1 0 0
1 0 0 1 1
1 0 0 0 1
0 1 0 0 1
0 1 1 1 0

```

Ilustración 13: Mostrar características del grafo.

3.3 VERIFICAR SI EL GRAFO ES CONEXO.

Esta funcionalidad verifica si un grafo es conexo, esto quiere decir que hay un camino entre todo par de vértices.

```
Ingrese una opcion: 3
El grafo es conexo
```

Ilustración 14: Verificar grafo, conexo.

```
Ingrese una opcion: 3
El grafo NO es conexo
```

Ilustración 15: Verificar grafo, desconexo.

3.4 ORDENAR POR CENTRALIDAD DE GRADO.

Esta funcionalidad ordena los vértices según la cantidad de vértices adyacentes para cada uno de ellos.

```
Ingrese una opcion: 4

Nodo 2: grado 3
Nodo 5: grado 3
Nodo 1: grado 2
Nodo 3: grado 2
Nodo 4: grado 2
```

Ilustración 16: Centralidad de grado.

3.5 ORDENAR POR CENTRALIDAD DE BETWEENNESS.

Esta funcionalidad del programa no está habilitada. Lo que realiza esta funcionalidad es ordenar los vértices según la centralidad de betweenness, lo que significa ordenar según el cociente entre la cantidad de veces que un nodo está presente entre los caminos mínimos entre dos nodos y el total de esos caminos mínimos.

CAPÍTULO 4. POSIBLES ERRORES.

El único error que se puede encontrar en el programa es que no funciona correctamente cuando se ingresa un grafo sin el formato predefinido anteriormente.

Cabe mencionar que la función de ordenar por centralidad de betweenness no esta habilitada.

Como no se encontraron más errores dentro del programa se enumerarán los errores solucionados:

1. Que existan espacios antes del final del salto de línea en el archivo de texto entregado.
2. Al ingresar un nombre de archivo inválido el programa sigue funcionando.