

# LAPORAN TUGAS KECIL 2 IF2211 STRATEGI ALGORITMA

## Semester II tahun 2020/2021

### **Penyusunan Rencana Kuliah dengan Topological Sort (Penerapan Decrease and Conquer)**

Faris Hasim Syauqi  
13519050

#### **A. Algoritma**

Berikut adalah langkah-langkah yang akan dilakukan dalam menyelesaikan persoalan penyusunan rencana kuliah dengan topological sort sebagai salah satu pendekatan algoritma *decrease and conquer*.

1. Ubahlah daftar mata kuliah beserta mata kuliah yang menjadi prasyaratnya ke dalam representasi graf. Graf yang dihasilkan haruslah merupakan *Directed Acyclic Graph* (DAG) yang merupakan graf berarah yang tidak mengandung satu pun siklus di dalamnya. Jika graf yang dihasilkan bukan merupakan DAG, maka persoalan tidak bisa diselesaikan dengan topological sort.
2. Pada DAG, pilihlah semua simpul yang memiliki derajat masuk bernilai 0. Untuk setiap simpul dengan derajat masuk 0, masukkan ke dalam list. List akan berisi semua mata kuliah yang dapat diambil pada semester ini. Untuk graf kosong, artinya tidak memiliki simpul, maka hasilnya adalah list kosong. Ini adalah kasus basis.
3. Untuk semua simpul yang dimasukkan ke dalam list, hapus simpul tersebut pada graf, beserta semua sisi yang keluar dari simpul tersebut.
4. Simpan list tersebut, kemudian ulangi langkah 2 secara rekursif untuk semester selanjutnya untuk memperoleh list mata kuliah yang dapat diambil pada semester-semester selanjutnya.
5. Satukan list mata kuliah yang bisa diambil untuk semester ini dengan list mata kuliah yang bisa diambil untuk semester-semester selanjutnya.

## Pendekatan *Decrease and Conquer*

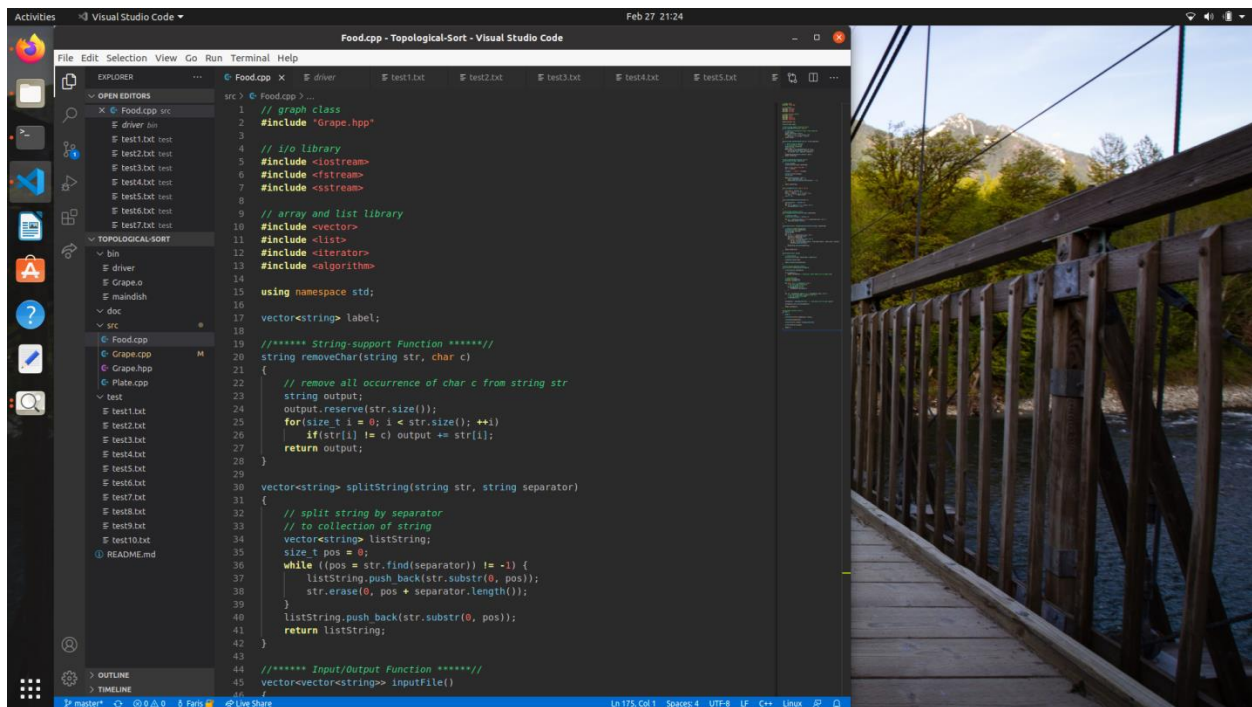
Decrease and conquer adalah sebuah metode perancangan algoritma untuk menyelesaikan suatu persoalan dengan cara mereduksi persoalan tersebut menjadi upa-persoalan yang lebih kecil, kemudian dilanjutkan dengan memproses upa-persoalan tersebut.

Pada Persoalan penyusunan rencana kuliah dengan topological sort ini, dapat dilihat bahwa algoritma yang digunakan merupakan salah satu jenis algoritma decrease and conquer, dimana untuk setiap iterasi-nya (semester), persoalan direduksi dengan cara menghilangkan semua mata kuliah yang dapat diambil untuk semester ini, yakni semua mata kuliah tanpa prasyarat, kemudian dilanjutkan dengan memproses upa-persoalan yang lebih kecil yakni penyusunan rencana kuliah untuk semester-semester selanjutnya secara rekursif.

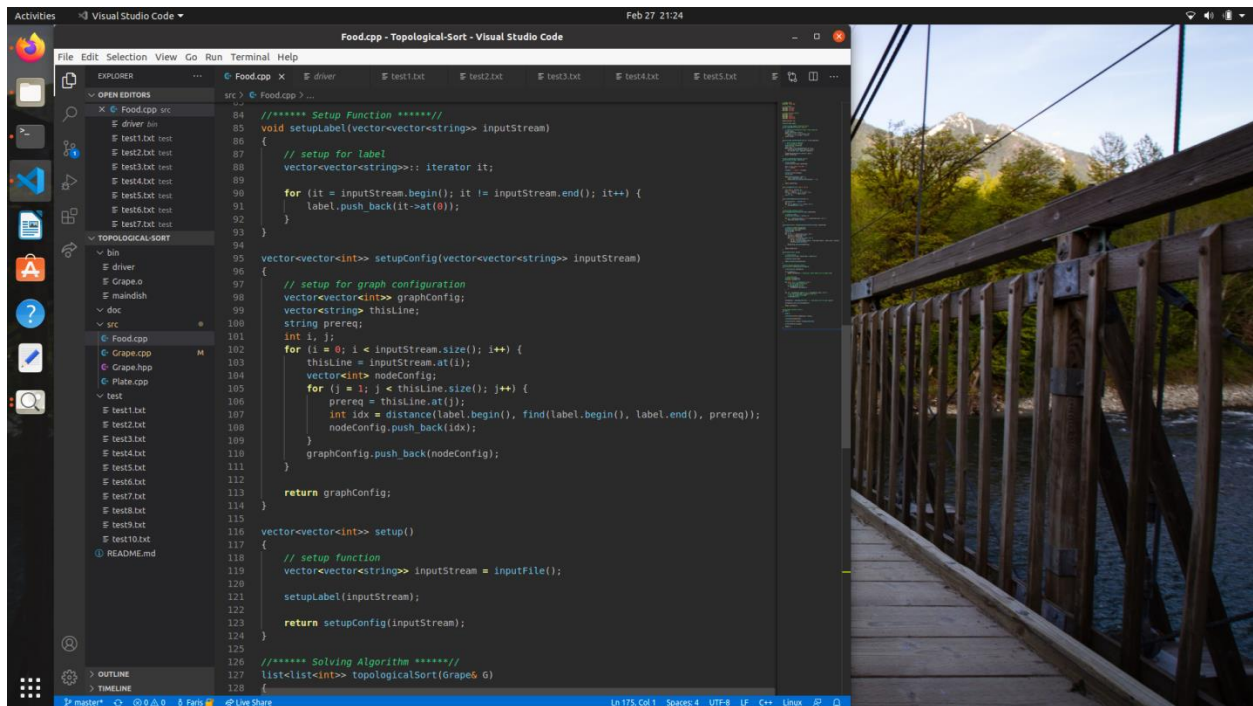
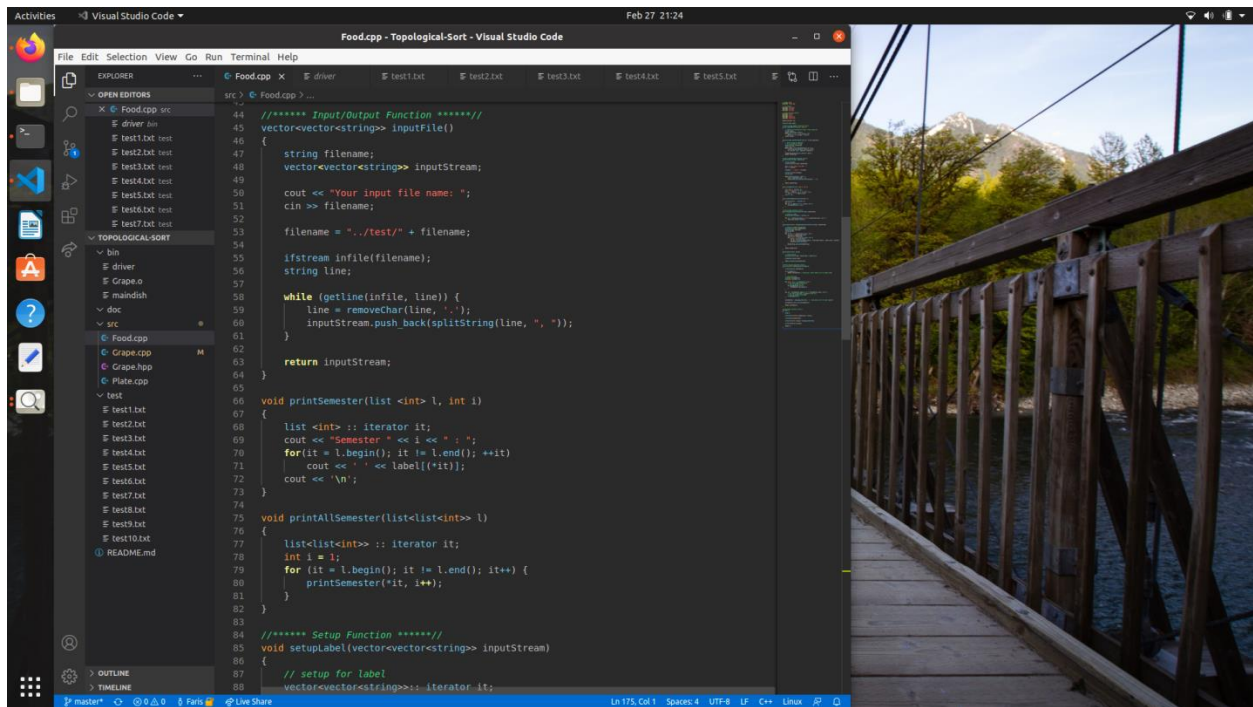
Penyelesaian persoalan penyusunan rencana kuliah dengan topological sort ini adalah salah satu algoritma decrease and conquer dengan jenis decrease by a variable size. Hal ini dikarenakan untuk setiap iterasinya, ukuran instan yang direduksi bervariasi tergantung pada banyaknya mata kuliah tanpa prasyarat di iterasi tersebut.

## B. Source Code Program

Berikut adalah *Source Code Program* Utama sebagai implementasi dari algoritma di atas.



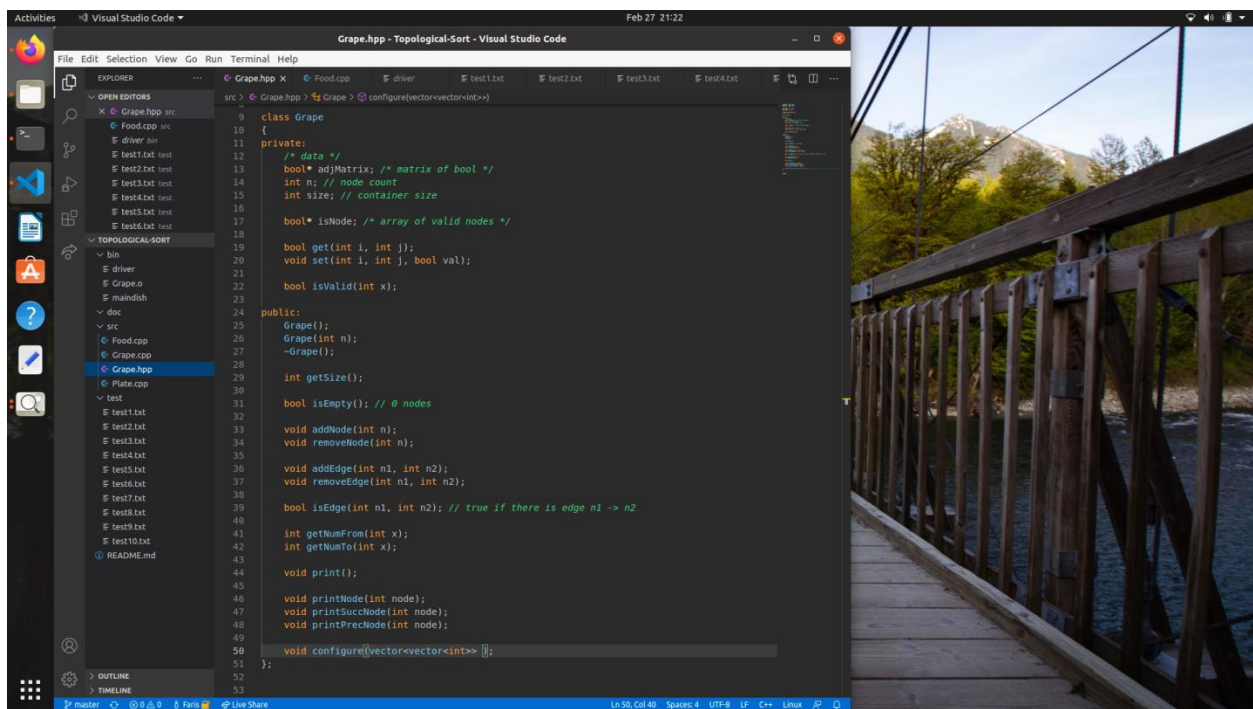
```
1 // graph class
2 #include "Grape.hpp"
3
4 // i/o library
5 #include <iostream>
6 #include <fstream>
7 #include <sstream>
8
9 // array and list library
10 #include <vector>
11 #include <list>
12 #include <iterator>
13 #include <algorithm>
14
15 using namespace std;
16
17 vector<string> label;
18
19 //***** String-support Function *****/
20 string removeChar(string str, char c)
21 {
22     // remove all occurrence of char c from string str
23     string output;
24     output.reserve(str.size());
25     for(size_t i = 0; i < str.size(); ++i)
26         if(str[i] != c) output += str[i];
27     return output;
28 }
29
30 vector<string> splitString(string str, string separator)
31 {
32     // split string by separator
33     // to collection of string
34     vector<string> listString;
35     size_t pos = 0;
36     while ((pos = str.find(separator)) != -1) {
37         listString.push_back(str.substr(0, pos));
38         str.erase(0, pos + separator.length());
39     }
40     listString.push_back(str.substr(0, pos));
41     return listString;
42 }
43
44 //***** Input/Output Function *****/
45 vector<vector<string>> inputFile()
```





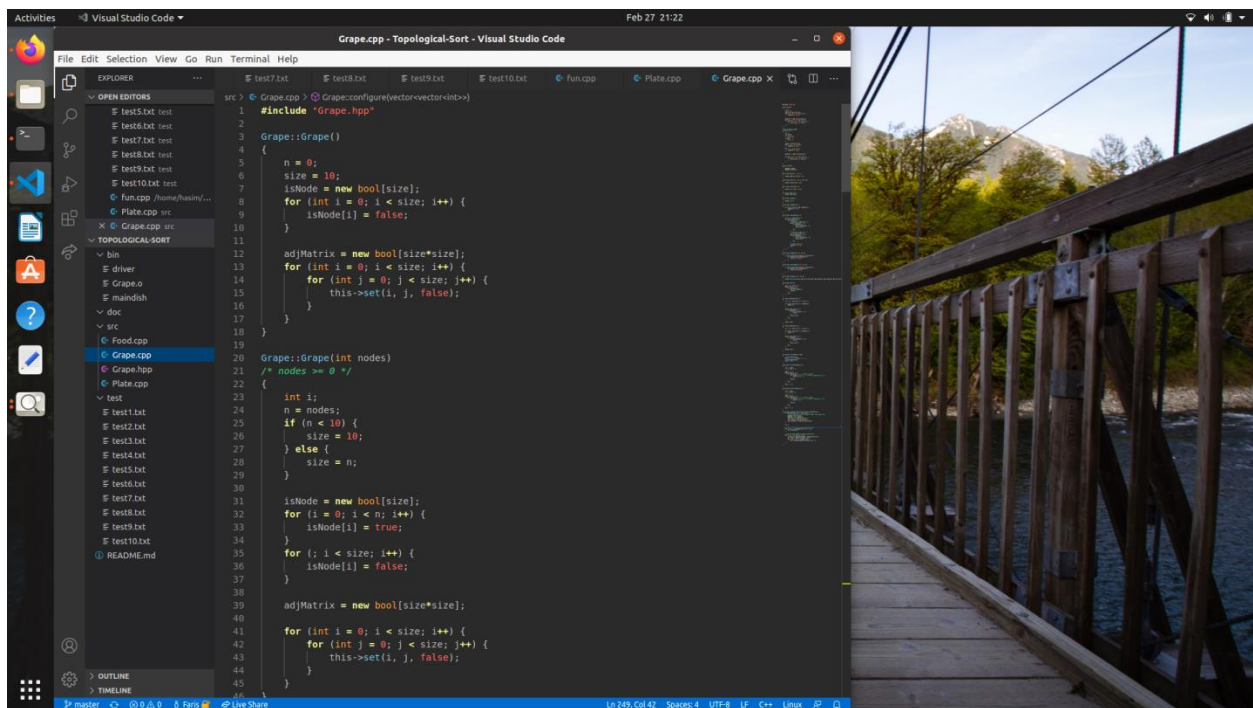


Berikut adalah *Source Code* header dan implementasi kelas graf yang digunakan pada program utama.



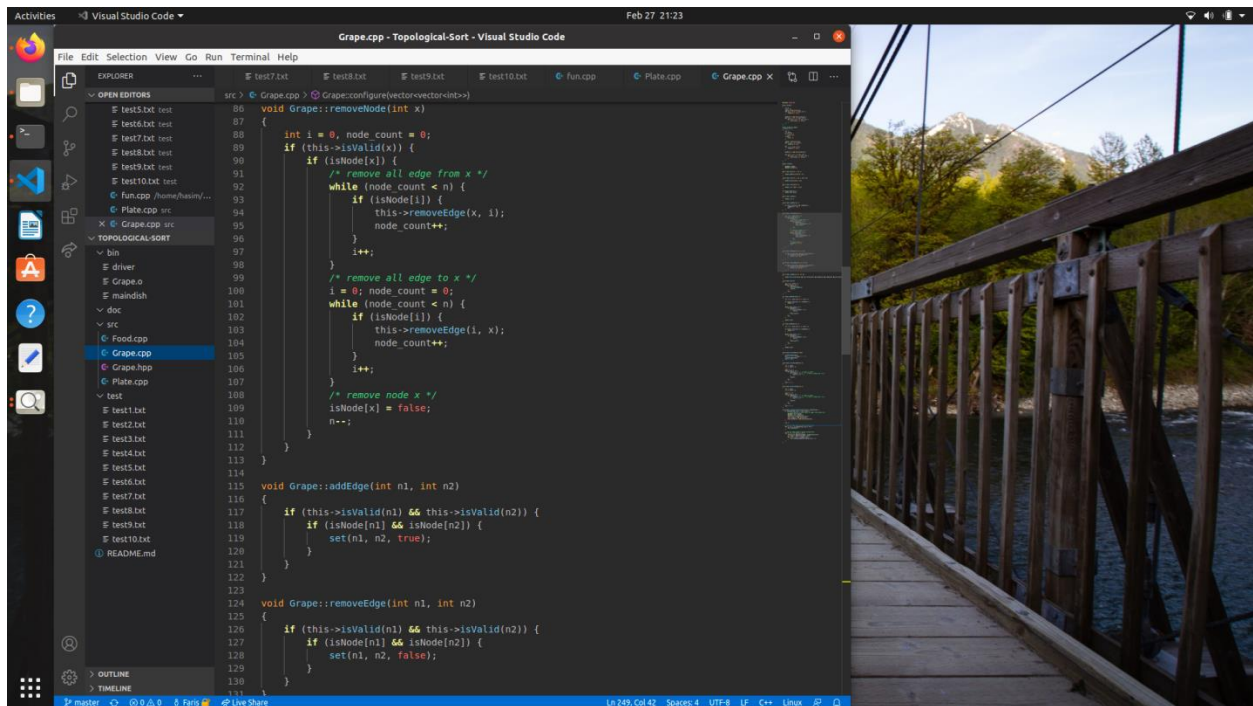
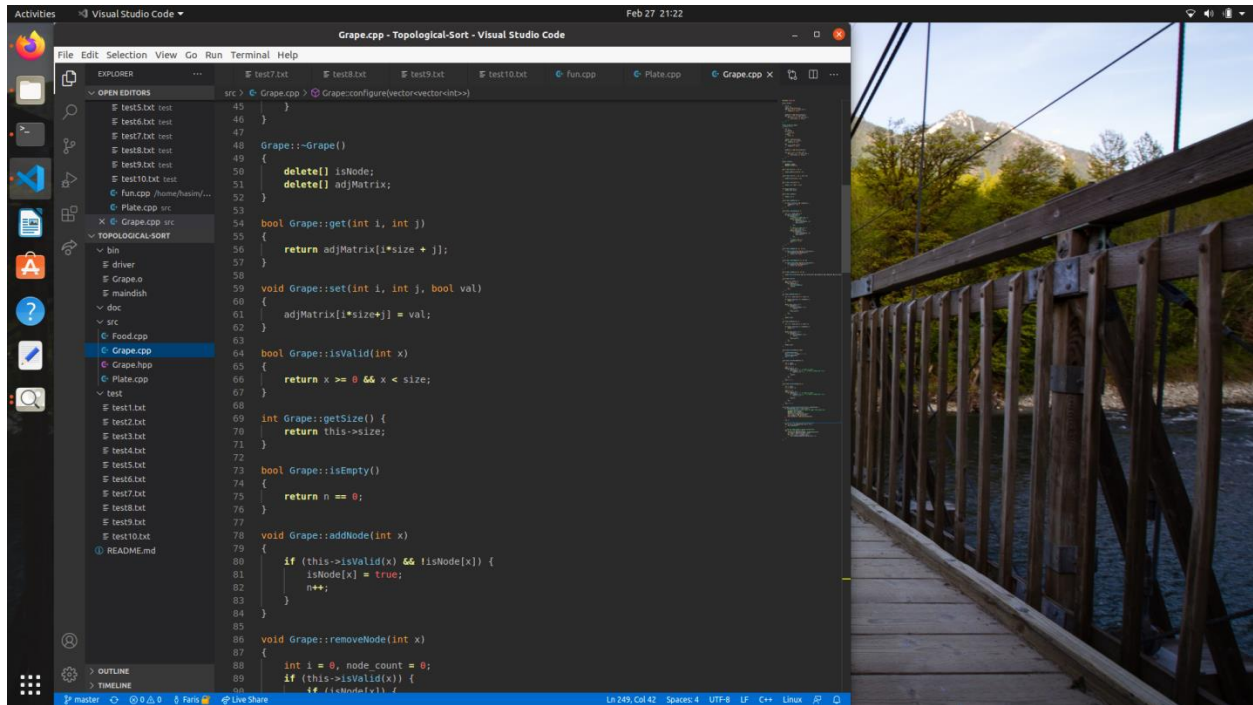
The screenshot shows the Visual Studio Code editor with the file `Grape.hpp` open. The code defines a `Grape` class with various methods for managing a graph. The Explorer sidebar on the left shows the project structure, including `src`, `bin`, `test`, and `doc` folders. The status bar at the bottom indicates the file is at line 50, column 40.

```
1  class Grape
2  {
3  private:
4      /* data */
5      bool* adjMatrix; /* matrix of bool */
6      int n; /* node count */
7      int size; /* container size */
8
9      bool* isNode; /* array of valid nodes */
10
11      bool get(int i, int j);
12      void set(int i, int j, bool val);
13
14      bool isValid(int x);
15
16 public:
17      Grape();
18      Grape(int n);
19      ~Grape();
20
21      int getSize();
22
23      bool isEmpty(); // 0 nodes
24
25      void addNode(int n);
26      void removeNode(int n);
27
28      void addEdge(int n1, int n2);
29      void removeEdge(int n1, int n2);
30
31      bool isEdge(int n1, int n2); // true if there is edge n1 -> n2
32
33      int getNumFrom(int x);
34      int getNumTo(int x);
35
36      void print();
37
38      void printNode(int node);
39      void printSuccNode(int node);
40      void printPrecNode(int node);
41
42      void configure(vector<vector<int>> &);
43
44
45
46
47
48
49
50
51
52
53 }
```

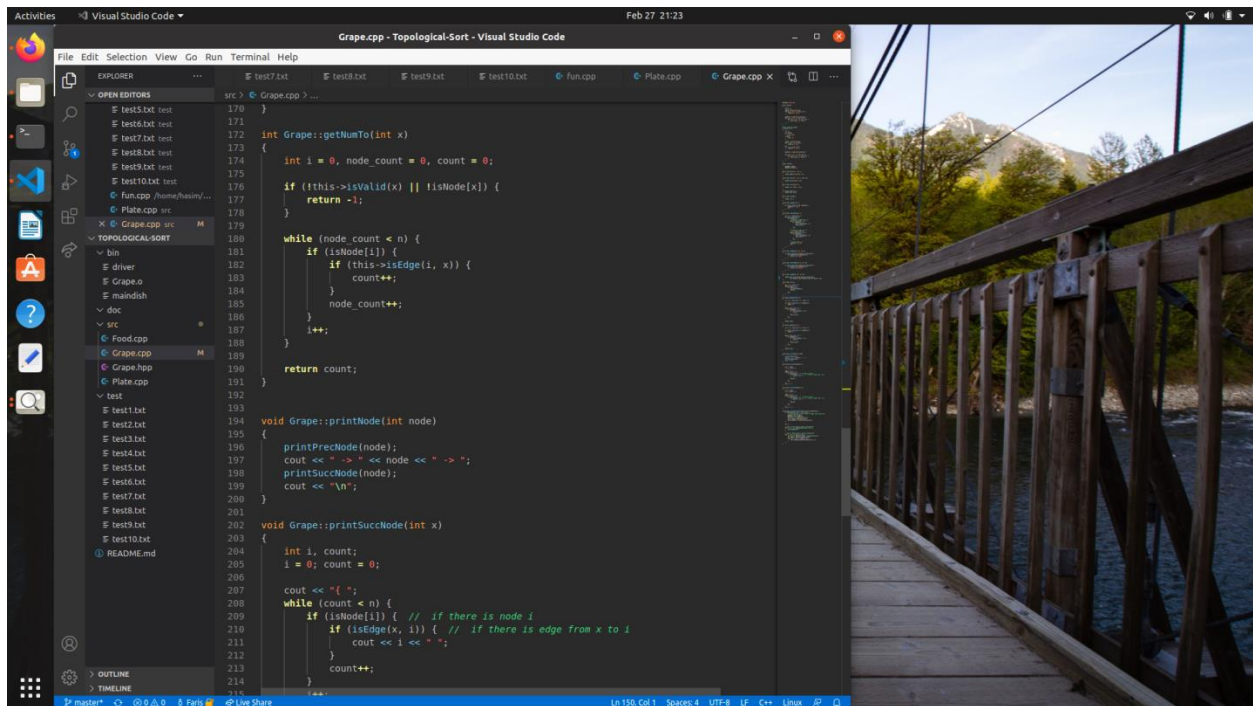
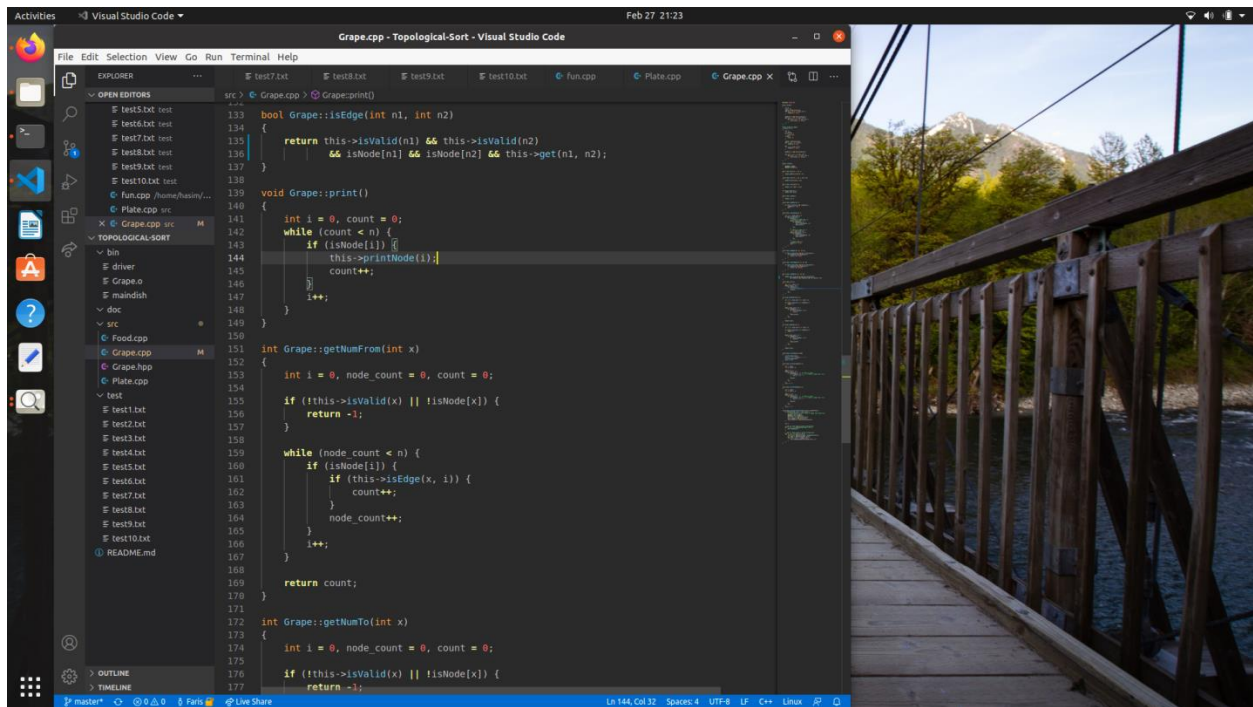


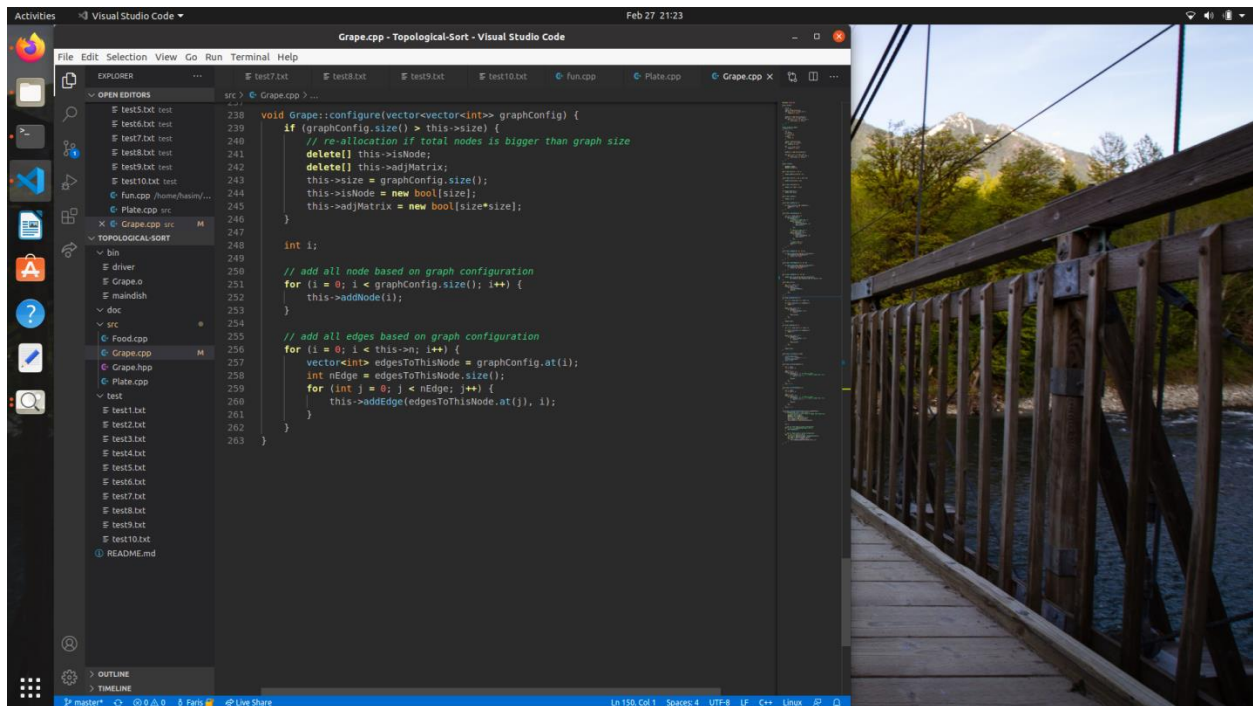
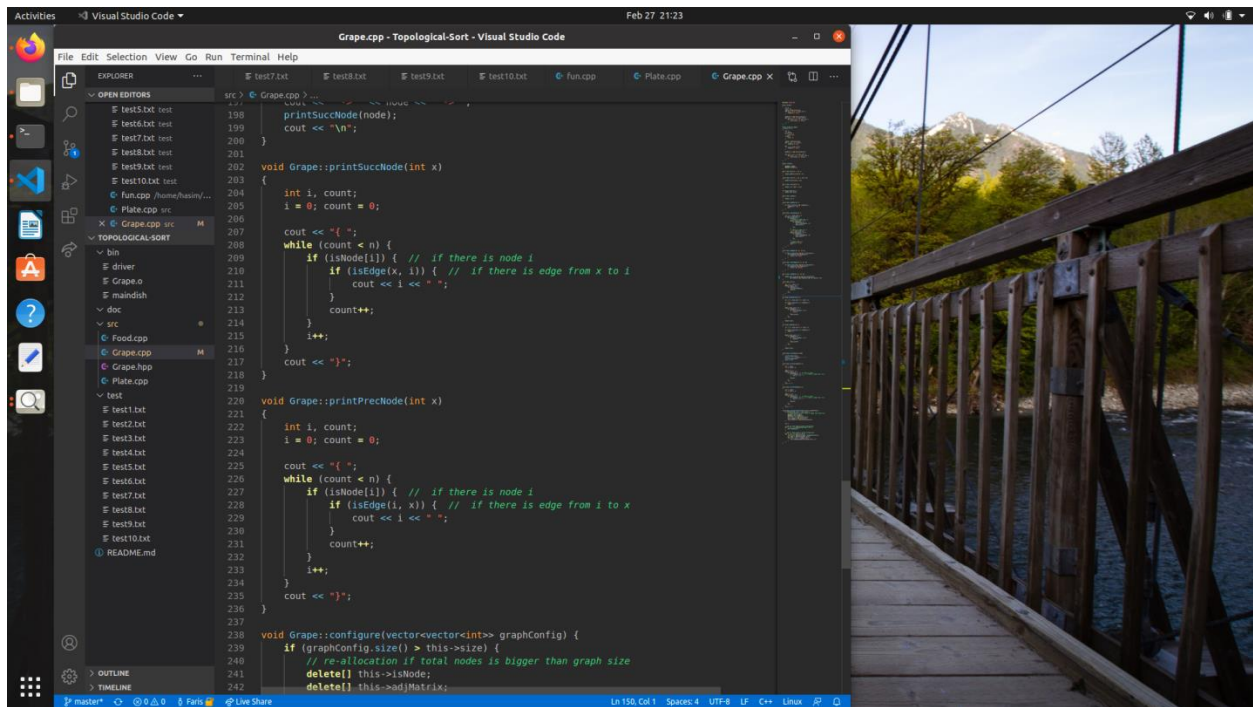
The screenshot shows the Visual Studio Code editor with the file `Grape.cpp` open. The code implements the methods defined in `Grape.hpp`. The Explorer sidebar on the left shows the project structure, including `src`, `bin`, `test`, and `doc` folders. The status bar at the bottom indicates the file is at line 249, column 42.

```
1  #include "Grape.hpp"
2
3  Grape::Grape()
4  {
5      n = 0;
6      size = 10;
7      isNode = new bool[size];
8      for (int i = 0; i < size; i++) {
9          isNode[i] = false;
10     }
11
12     adjMatrix = new bool[size*size];
13     for (int i = 0; i < size; i++) {
14         for (int j = 0; j < size; j++) {
15             this->set(i, j, false);
16         }
17     }
18 }
19
20 Grape::Grape(int nodes)
21 /* nodes >= 0 */
22 {
23     int i;
24     n = nodes;
25     if (n < 10) {
26         size = 10;
27     } else {
28         size = n;
29     }
30
31     isNode = new bool[size];
32     for (i = 0; i < n; i++) {
33         isNode[i] = true;
34     }
35     for (; i < size; i++) {
36         isNode[i] = false;
37     }
38
39     adjMatrix = new bool[size*size];
40
41     for (int i = 0; i < size; i++) {
42         for (int j = 0; j < size; j++) {
43             this->set(i, j, false);
44         }
45     }
46 }
```





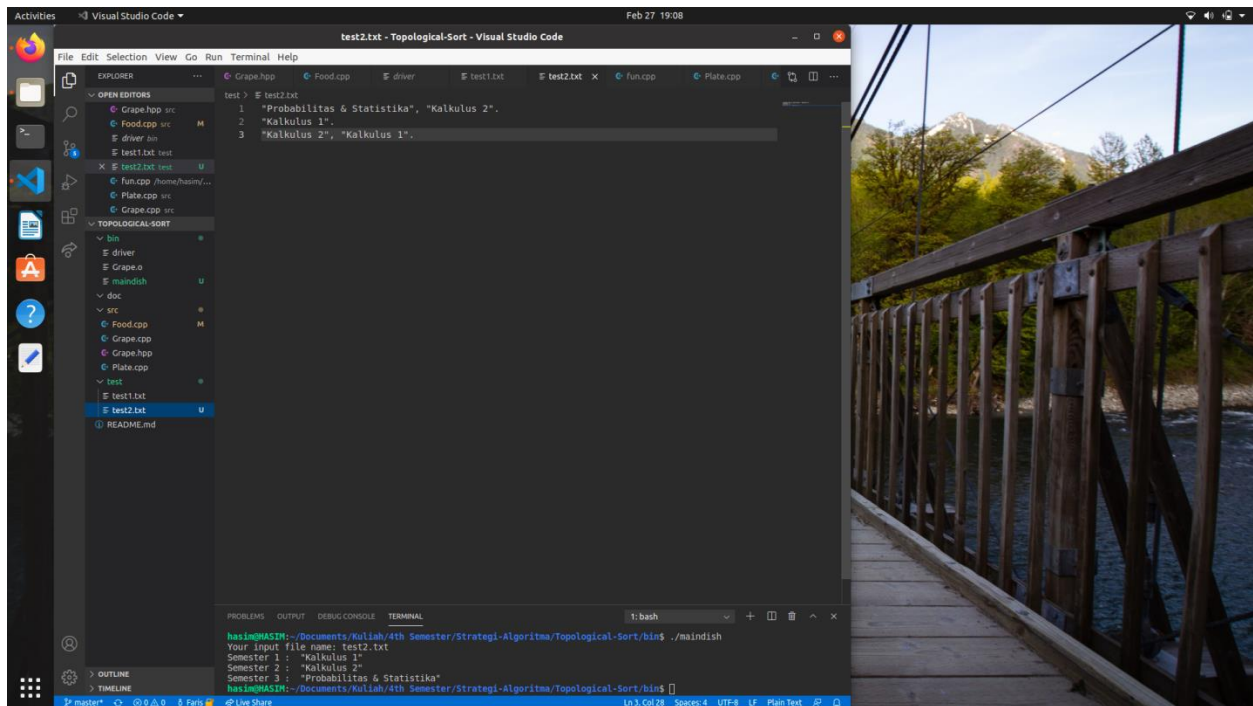
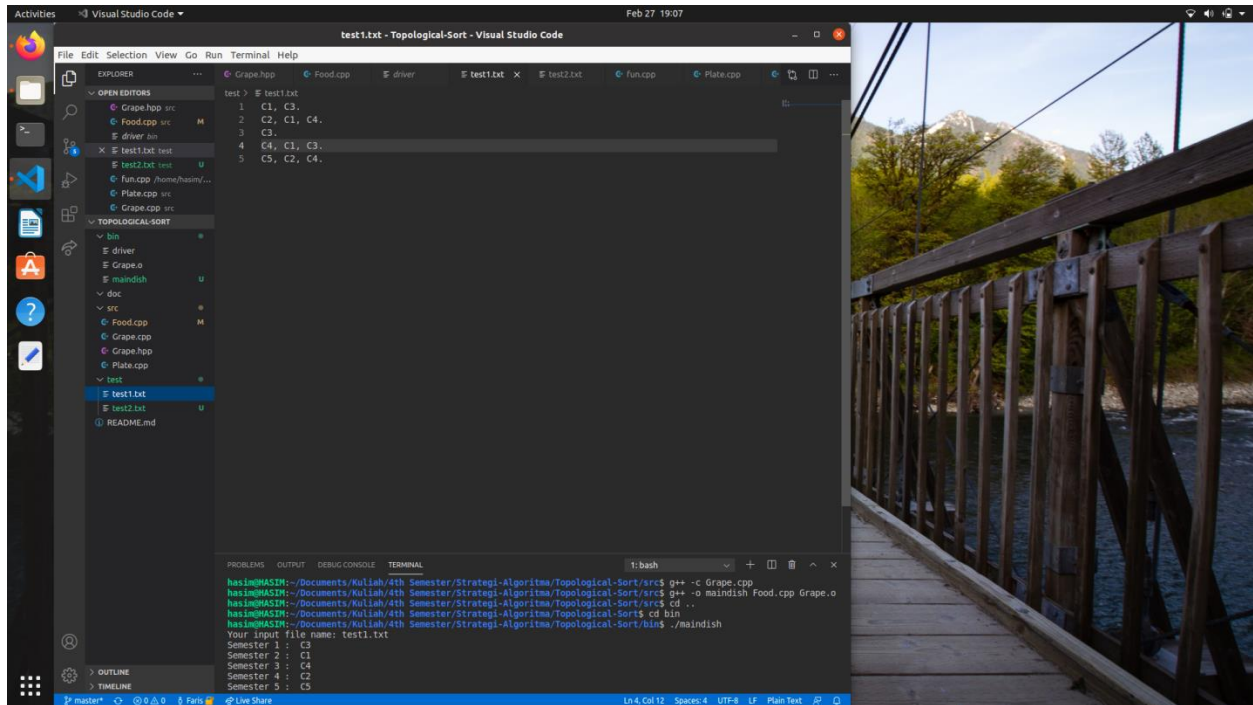


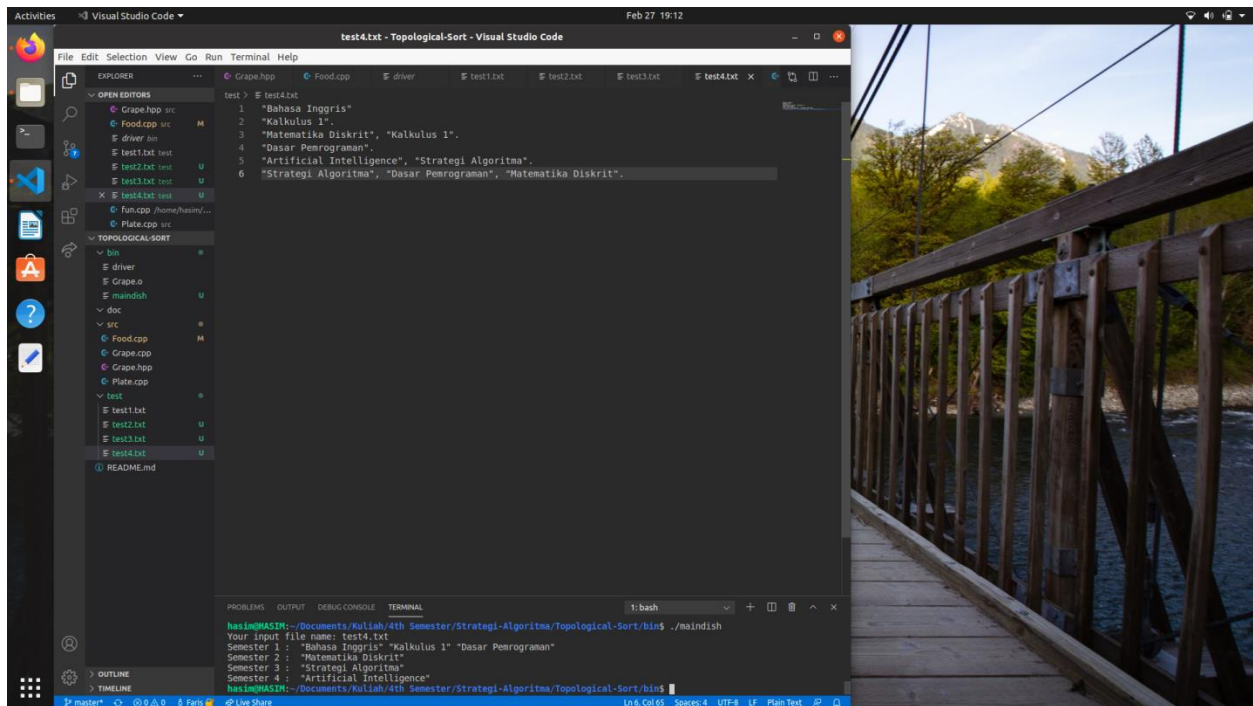
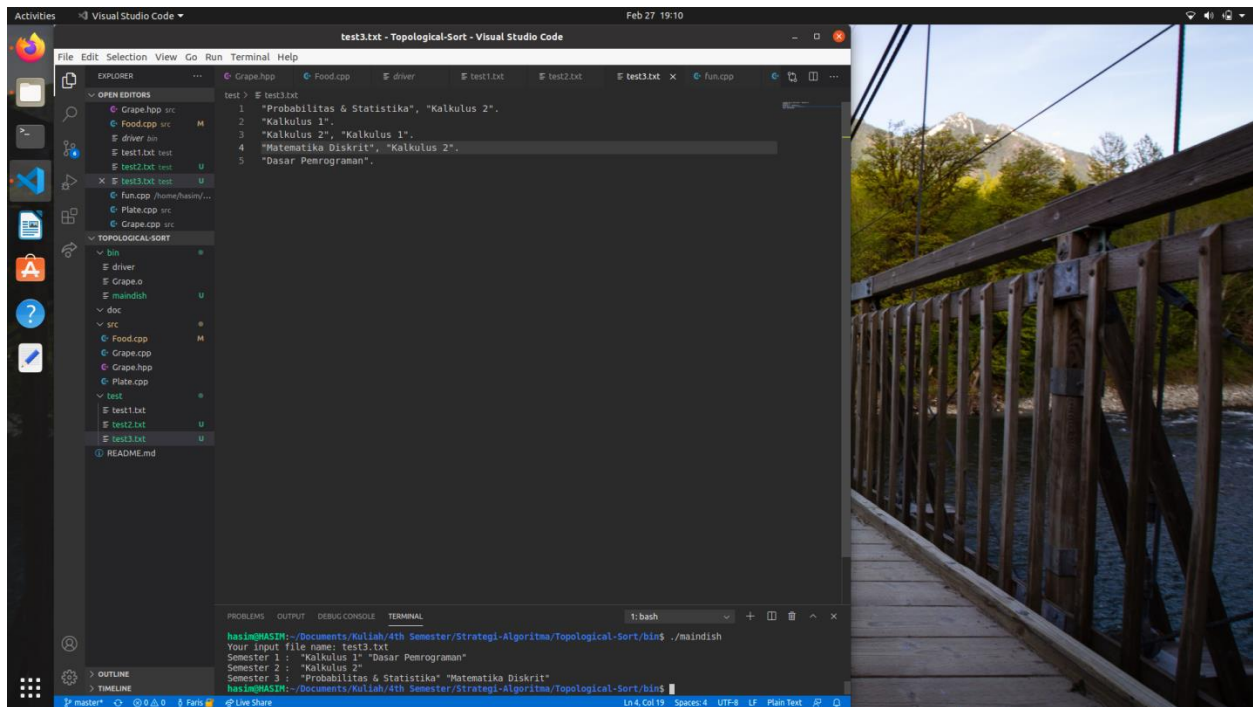


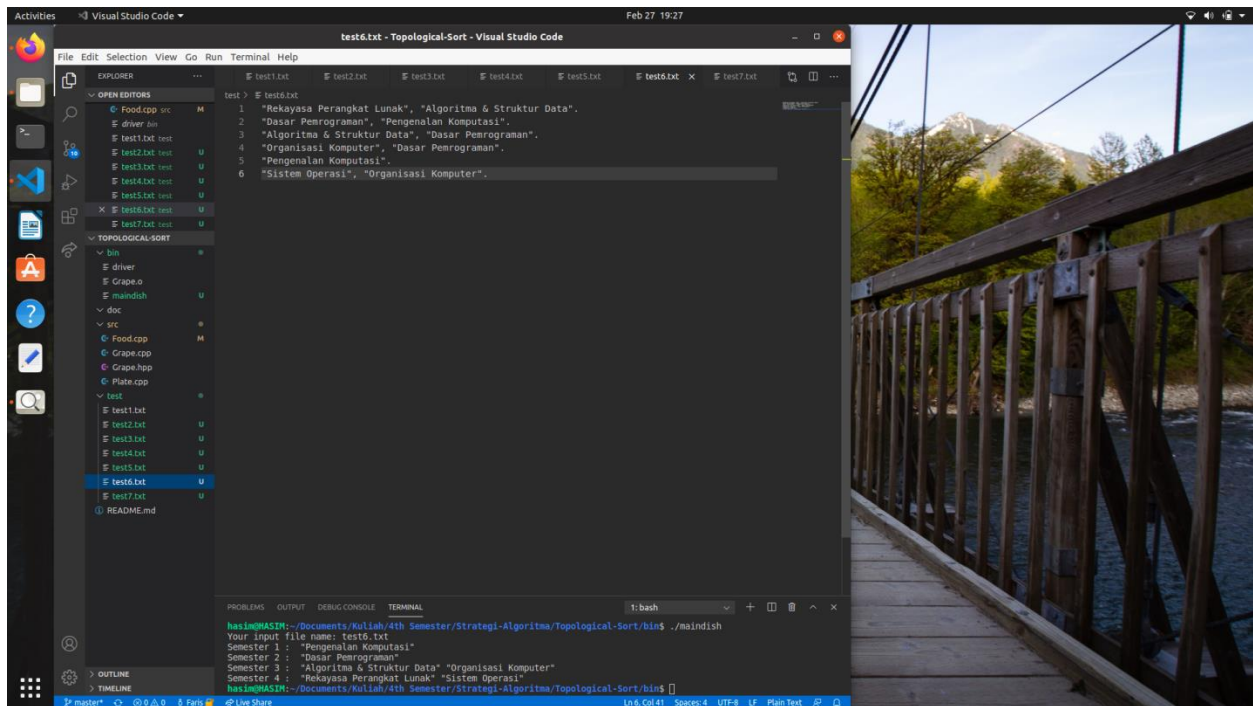
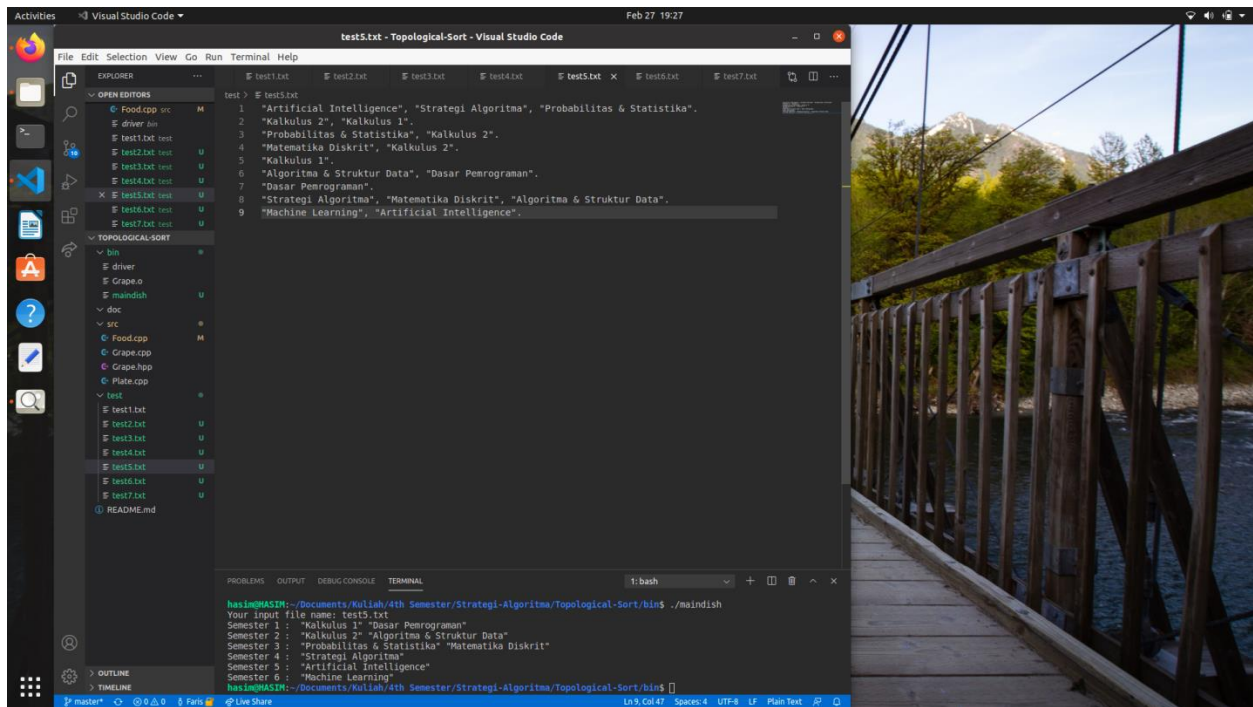


## C. Screenshot Percobaan

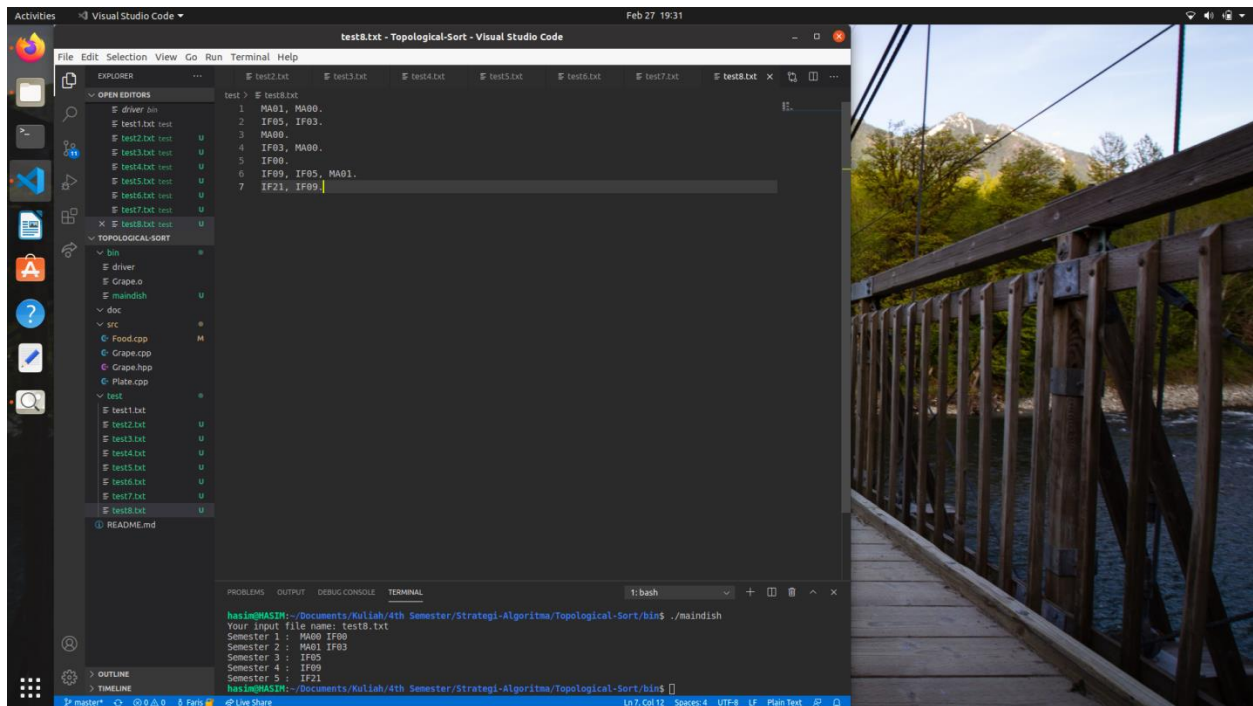
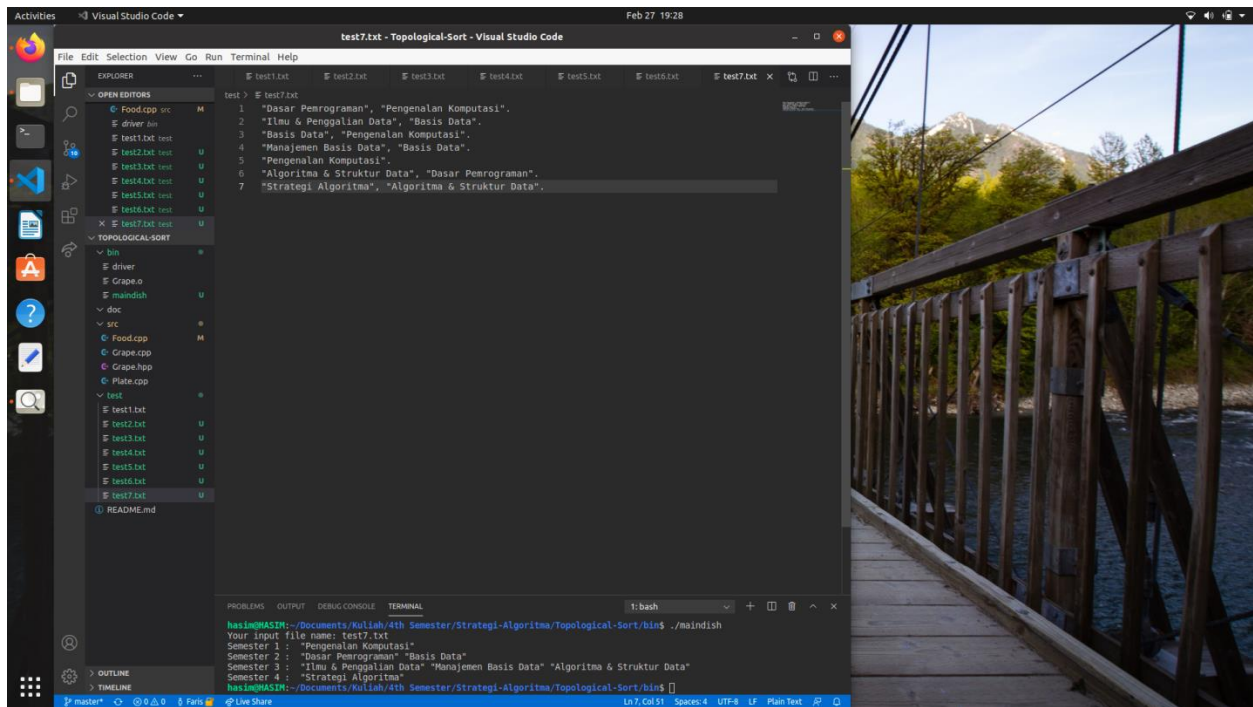
Berikut adalah beberapa foto hasil percobaan dari program.

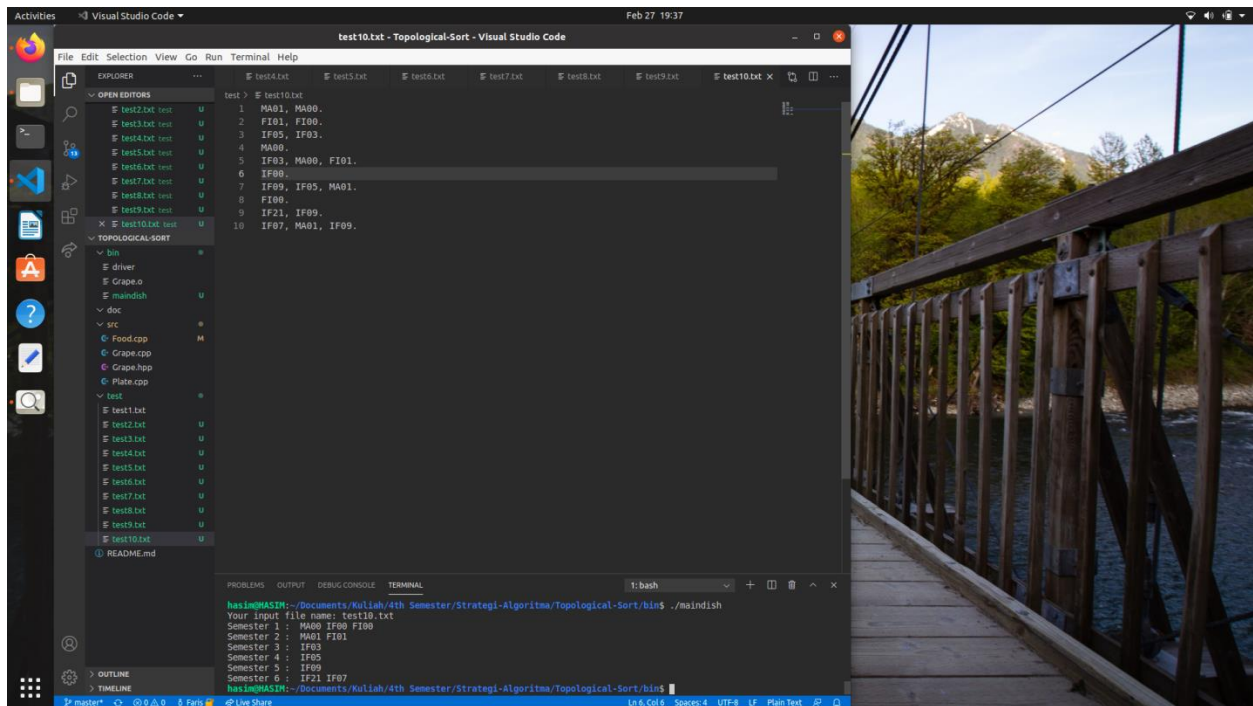
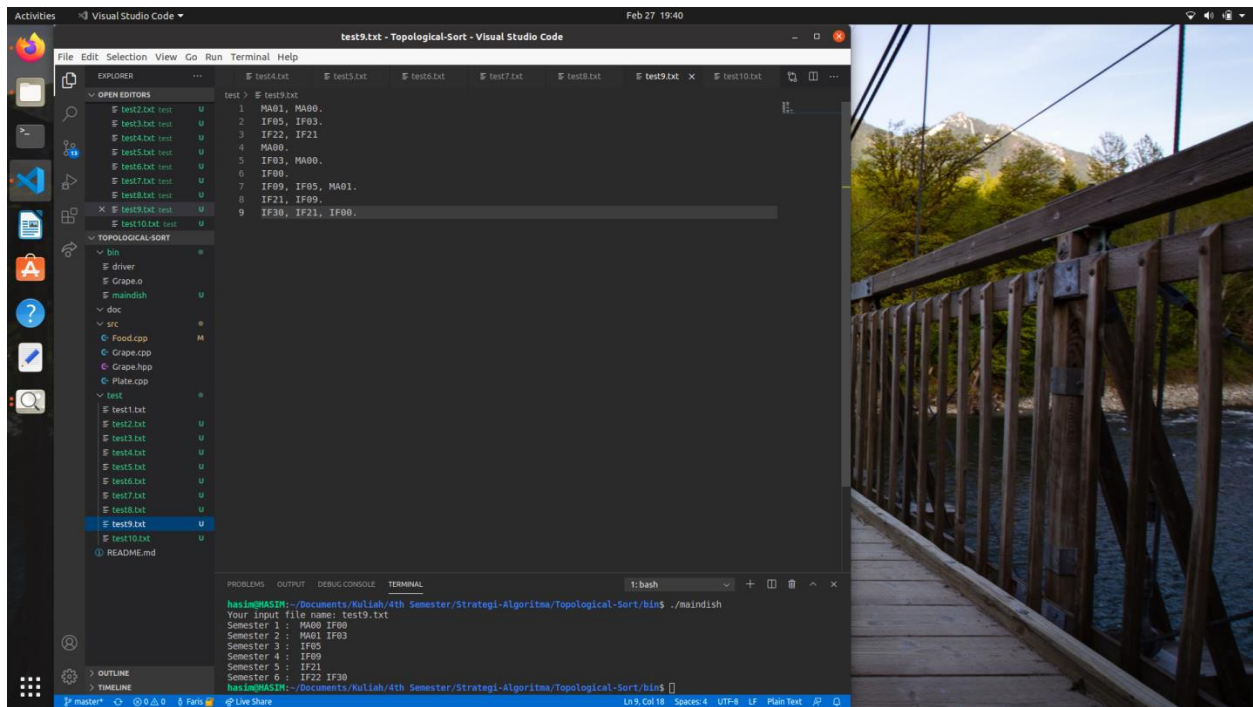












#### **D. Alamat Kode Program**

Program dapat diunduh dari github pada tautan berikut :

GitHub : <https://github.com/farishasim/Topological-Sort>

#### **E. Tabel Penilaian**

<b>Poin</b>	<b>Ya</b>	<b>Tidak</b>
1. Program berhasil dikompilasi	√	
2. Program berhasil running	√	
3. Program dapat menerima berkas input dan menuliskan output.	√	
4. Luaran sudah benar untuk semua kasus input.	√	