

# HW#4 Graph 구현

2024.05.30 (목) 09:00 ~ 2024.06.19 (수) 23:59

## 00. 과제 주의사항

- "FILL YOUR CODE" 주석과 "YOUR CODE ENDS HERE" 주석 **사이에만** 코드를 작성하세요.
- "FILL YOUR CODE" 주석과 "YOUR CODE ENDS HERE" 주석을 **삭제하지** 마세요.
- 제공되는 코드의 **변형/추가/삭제**는 허용하지 않습니다.
- 제출 파일 및 제출 양식 **정확하게** 지켜주세요.

```
/* 파일로부터 트리를 구성하는 함수 */
bool construct_tree_from_file(BST* Tree, char* filename) {
    /*
    INPUT:
        BST* Tree: 트리 구조체 포인터
        char* filename: 파일 이름

    OUTPUT:
        bool: 파일 열기 성공 여부
        true: 파일 열기 성공
        false: 파일 열기 실패

    DESCRIPTION:
        이 함수는 파일로부터 트리를 구성하는 함수입니다.
        파일을 열어서 학생 이름과 학번을 읽어서 트리에 추가합니다.
        파일 열기에 성공하면 true를 반환하고, 실패하면 false를 반환합니다.
    */

    FILE* fp = fopen(filename, "r");
    if (!fp) return false;

    char std_name[50];
    int std_num;

    /* ===== FILL YOUR CODE ===== */
    여기에 코드 작성
    /* ===== YOUR CODE ENDS HERE ===== */

    fclose(fp);
    return true;
}
```

- 자동 채점 프로그램의 도움을 받아 채점합니다.
  - 프로그램의 출력 양식 변화, 제공된 코드 및 주석에 변화가 생기면 채점 시 **불이익**이 있습니다.
  - 과제로 구현해야 하는 코드에는 **출력문**을 작성할 필요가 없습니다.

## 01. 과제 목표

---

- Graph 구현 및 기본 연산의 작동 원리 이해
  - Graph 구현
- 그래프 순회 알고리즘, 최단경로 알고리즘 구현 및 작동 원리 이해
  - DFS
  - BFS
  - Dijkstra
  - Bellman\_ford
  - Floyd\_warshall

## 02. 과제 준비

- Visual Studio (버전 2019 또는 2022)로 작성해야 합니다.
  - IDE 불일치에 따른 오류는 책임지지 않습니다.

- 아래의 4가지 파일은 **프로젝트 디렉토리** 안에 있어야 합니다.

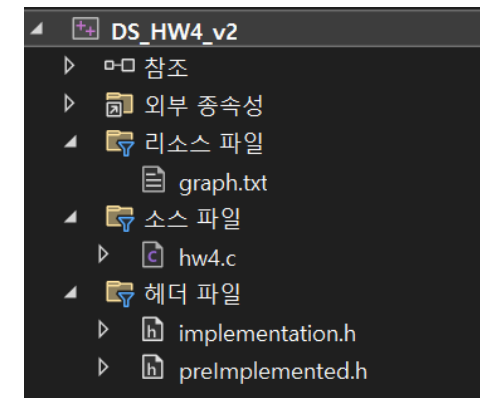
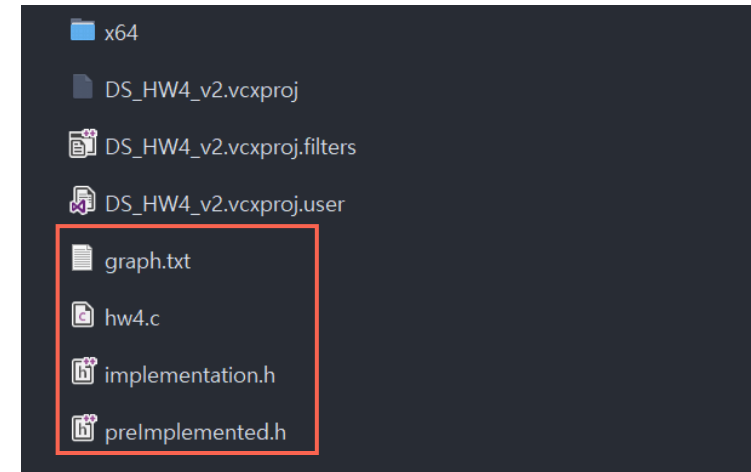
- graph.txt
  - 그래프 데이터가 들어있습니다.
- hw4.c
  - main 함수가 동작합니다.
- preImplemented.h
  - 미리 구현된 구조체 및 함수들이 있는 헤더 파일입니다.
  - **과제 진행을 위해 읽어 봐야 합니다.**
- implementation.h
  - 구현해야 하는 함수를 정의하는 헤더 파일입니다.
  - **과제를 직접적으로 진행하게 될 파일입니다.**

- Visual Studio 솔루션 탐색기에서 다음 사진과 같이 각 소스를 잡아주세요.

- [리소스 파일/소스 파일/헤더 파일] 우클릭 > 추가 > 기존 항목

- **프로젝트 디렉토리**

"C:/Users/윈도우사용자명/source/repos/프로젝트명/프로젝트명/"



## 02. 과제 준비

- 그래프 정보는 data.txt 파일로 제공됩니다.
- 해당 파일의 포맷은 다음과 같습니다.
  - {시작정점} {종료정점} {가중치}
- 프로그램 시작 시 텍스트 파일의 이름과 그래프 타입 (방향, 무방향)을 넣어줘야 합니다.

```
C:\Users\WshigGy\sourceWre x + v
그래프 데이터 파일 이름 입력 : graph.txt
그래프 종류 (0: 무방향, 1: 방향) : 0
>>>
```

0	1	4
0	3	1
1	2	2
2	4	7
3	4	3
5	6	2
6	7	1
7	5	4
8	9	5
9	4	1
4	8	2
0	5	3
2	6	5
3	7	6
1	9	2

## 03. 과제 내용

- 구현해야 하는 함수는 총 11개입니다.

```
/* 함수 프로토타입 선언 (구현해야 하는 함수들입니다.) */  
bool construct_graph_from_file(Graph* graph, FILE* fp);  
bool insert_node(Graph* graph, int node);  
bool delete_node(Graph* graph, int node);  
bool insert_edge(Graph* graph, int v1, int v2, double cost);  
bool delete_edge(Graph* graph, int v1, int v2);  
void DFS_iterative(Graph* graph, int start_node);  
void DFS_recursive(Graph* graph, int node, bool visited[]);  
void BFS(Graph* graph, int start_node);  
PathInfo dijkstra(Graph* graph, int start, int end);  
PathInfo bellman_ford(Graph* graph, int start, int end);  
PathInfo floyd_warshall(Graph* graph);
```

### 03. 과제 내용

- 프로그램이 사용하는 명령어는 다음과 같습니다.

#### 1. node

- node ins  $v$
- node del  $v$

무방향 그래프 사용 추천

#### 2. edge

- edge ins  $v_1 v_2 c$
- edge del  $v_1 v_2$

#### 3. dfs $v_s$

#### 4. bfs $v_s$

#### 5. dij $v_s v_e$

#### 6. befo $v_s v_3$

방향 그래프 사용 추천

#### 7. flwa

#### 8. print

#### 9. exit

## 03. 과제 내용

- node ins  $v$ 
  - 그래프에 노드  $v$  를 추가합니다.

```

C:\Users\Shiggy\source\Wre x + v
그래프 데이터 파일 이름 입력 : graph.txt
그래프 종류 (0: 무방향, 1: 방향) : 0
>>> node ins 10
Directed Graph: False
Number of Nodes: 11
0 1 2 3 4 5 6 7 8 9 10
Adjacency List:
0: (1, 4.0) (3, 1.0) (5, 3.0)
1: (0, 4.0) (2, 2.0) (9, 2.0)
2: (1, 2.0) (4, 7.0) (6, 5.0)
3: (0, 1.0) (4, 3.0) (7, 6.0)
4: (2, 7.0) (3, 3.0) (8, 2.0) (9, 1.0)
5: (0, 3.0) (6, 2.0) (7, 4.0)
6: (2, 5.0) (5, 2.0) (7, 1.0)
7: (3, 6.0) (5, 4.0) (6, 1.0)
8: (4, 2.0) (9, 5.0)
9: (1, 2.0) (4, 1.0) (8, 5.0)
10:
Adjacency Matrix:
INF 4.0 INF 1.0 INF 3.0 INF INF INF INF INF INF INF INF
4.0 INF 2.0 INF INF INF INF INF 2.0 INF INF INF INF INF
INF 2.0 INF INF 7.0 INF 5.0 INF INF INF INF INF INF INF
1.0 INF INF INF 3.0 INF INF 6.0 INF INF INF INF INF INF
INF INF 7.0 3.0 INF INF INF 2.0 1.0 INF INF INF INF INF
3.0 INF INF INF INF INF 2.0 4.0 INF INF INF INF INF INF
INF INF 5.0 INF INF 2.0 INF 1.0 INF INF INF INF INF
INF INF INF 6.0 INF 4.0 1.0 INF INF INF INF INF INF
INF INF INF INF 2.0 INF INF INF 5.0 INF INF INF INF
INF 2.0 INF INF 1.0 INF INF 5.0 INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF
>>>

```

- node del  $v$ 
  - 그래프에서 노드  $v$  를 삭제합니다.
    - 기존 노드  $v$  와 연결되어 있던 edge 모두 삭제되어야 합니다.

```

C:\Users\Shiggy\source\Wre x + v
그래프 데이터 파일 이름 입력 : graph.txt
그래프 종류 (0: 무방향, 1: 방향) : 0
>>> node del 3
Directed Graph: False
Number of Nodes: 9
0 1 2 4 5 6 7 8 9
Adjacency List:
0: (1, 4.0) (5, 3.0)
1: (0, 4.0) (2, 2.0) (9, 2.0)
2: (1, 2.0) (4, 7.0) (6, 5.0)
4: (2, 7.0) (8, 2.0) (9, 1.0)
5: (0, 3.0) (6, 2.0) (7, 4.0)
6: (2, 5.0) (5, 2.0) (7, 1.0)
7: (5, 4.0) (6, 1.0)
8: (4, 2.0) (9, 5.0)
9: (1, 2.0) (4, 1.0) (8, 5.0)
Adjacency Matrix:
INF 4.0 INF INF INF 3.0 INF INF INF INF INF INF INF
4.0 INF 2.0 INF INF INF INF INF 2.0 INF INF INF INF
INF 2.0 INF INF 7.0 INF 5.0 INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF
INF INF 7.0 INF INF INF INF 2.0 1.0 INF INF INF INF
3.0 INF INF INF INF INF 2.0 4.0 INF INF INF INF INF
INF INF 5.0 INF INF 2.0 INF 1.0 INF INF INF INF
INF INF INF INF INF 4.0 1.0 INF INF INF INF INF
INF INF INF INF 2.0 INF INF INF 5.0 INF INF INF
INF 2.0 INF INF 1.0 INF INF INF 5.0 INF INF INF
INF INF INF INF INF INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF
>>>

```



### 03. 과제 내용

- `edge ins  $v_1$   $v_2$   $c$` 
  - 그래프에  $(v_1, v_2, c)$  간선을 추가합니다.

```
C:\Users\ShigGy\source\Wre x + v
그래프 데이터 파일 이름 입력 : graph.txt
그래프 종류 (0: 무방향, 1: 방향) : 0
>>> edge ins 2 5 19.0
Directed Graph: False
Number of Nodes: 10
0 1 2 3 4 5 6 7 8 9
Adjacency List:
0: (1, 4.0) (3, 1.0) (5, 3.0)
1: (0, 4.0) (2, 2.0) (9, 2.0)
2: (1, 2.0) (4, 7.0) (5, 19.0) (6, 5.0)
3: (0, 1.0) (4, 3.0) (7, 6.0)
4: (2, 7.0) (3, 3.0) (8, 2.0) (9, 1.0)
5: (0, 3.0) (2, 19.0) (6, 2.0) (7, 4.0)
6: (2, 5.0) (5, 2.0) (7, 1.0)
7: (3, 6.0) (5, 4.0) (6, 1.0)
8: (4, 2.0) (9, 5.0)
9: (1, 2.0) (4, 1.0) (8, 5.0)
Adjacency Matrix:
INF 4.0 INF 1.0 INF 3.0 INF INF INF INF INF INF INF
4.0 INF 2.0 INF INF INF INF INF 2.0 INF INF INF INF
INF 2.0 INF INF 7.0 19.0 5.0 INF INF INF INF INF INF
1.0 INF INF INF 3.0 INF INF 6.0 INF INF INF INF INF
INF INF 7.0 3.0 INF INF INF 2.0 1.0 INF INF INF INF
3.0 INF 19.0 INF INF INF 2.0 4.0 INF INF INF INF INF
INF INF 5.0 INF INF 2.0 INF 1.0 INF INF INF INF INF
INF INF INF 6.0 INF 4.0 1.0 INF INF INF INF INF INF
INF INF INF INF 2.0 INF INF 5.0 INF INF INF INF INF
INF 2.0 INF INF 1.0 INF INF 5.0 INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF
>>>
```

- `edge del  $v_1$   $v_2$` 
  - 그래프에서 간선  $(v_1, v_2)$ 를 삭제합니다.

```
C:\Users\ShigGy\source\Wre x + v
INF 2.0 INF INF 1.0 INF INF INF 5.0 INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF INF
>>> edge del 2 5
Directed Graph: False
Number of Nodes: 10
0 1 2 3 4 5 6 7 8 9
Adjacency List:
0: (1, 4.0) (3, 1.0) (5, 3.0)
1: (0, 4.0) (2, 2.0) (9, 2.0)
2: (1, 2.0) (4, 7.0) (6, 5.0)
3: (0, 1.0) (4, 3.0) (7, 6.0)
4: (2, 7.0) (3, 3.0) (8, 2.0) (9, 1.0)
5: (0, 3.0) (6, 2.0) (7, 4.0)
6: (2, 5.0) (5, 2.0) (7, 1.0)
7: (3, 6.0) (5, 4.0) (6, 1.0)
8: (4, 2.0) (9, 5.0)
9: (1, 2.0) (4, 1.0) (8, 5.0)
Adjacency Matrix:
INF 4.0 INF 1.0 INF 3.0 INF INF INF INF INF INF INF
4.0 INF 2.0 INF INF INF INF INF 2.0 INF INF INF INF
INF 2.0 INF INF 7.0 INF 5.0 INF INF INF INF INF INF
1.0 INF INF INF 3.0 INF INF 6.0 INF INF INF INF INF
INF INF 7.0 3.0 INF INF INF 2.0 1.0 INF INF INF INF
3.0 INF 19.0 INF INF INF 2.0 4.0 INF INF INF INF INF
INF INF 5.0 INF INF 2.0 INF 1.0 INF INF INF INF INF
INF INF INF 6.0 INF 4.0 1.0 INF INF INF INF INF INF
INF INF INF INF 2.0 INF INF 5.0 INF INF INF INF INF
INF 2.0 INF INF 1.0 INF INF 5.0 INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF
INF INF INF INF INF INF INF INF INF INF INF INF
>>>
```

### 03. 과제 내용

- dfs  $v_s$ 
  - $v_s$  로부터 시작하는 DFS 알고리즘을 수행합니다.

```

C:\Users\ShigGy\source\Wre × + ∨
그래프 데이터 파일 이름 입력 : graph.txt
그래프 종류 (0: 무방향, 1: 방향) : 0
>>> dfs 3
반복문 버전 DFS 탐색 순서 :
3 0 1 2 4 8 9 6 5 7
재귀함수 버전 DFS 탐색 순서 :
3 0 1 2 4 8 9 6 5 7
>>> dfs 1
반복문 버전 DFS 탐색 순서 :
1 0 3 4 2 6 5 7 8 9
재귀함수 버전 DFS 탐색 순서 :
1 0 3 4 2 6 5 7 8 9
>>> dfs 2
반복문 버전 DFS 탐색 순서 :
2 1 0 3 4 8 9 7 5 6
재귀함수 버전 DFS 탐색 순서 :
2 1 0 3 4 8 9 7 5 6
>>> |

```

- bfs  $v_s$ 
  - $v_s$  로부터 시작하는 BFS 알고리즘을 수행합니다.

```

C:\Users\ShigGy\source\Wre × + ∨
그래프 데이터 파일 이름 입력 : graph.txt
그래프 종류 (0: 무방향, 1: 방향) : 0
>>> bfs 2
BFS 탐색 순서 :
2 1 4 6 0 9 3 8 5 7
>>> bfs 1
BFS 탐색 순서 :
1 0 2 9 3 5 4 6 8 7
>>> bfs 9
BFS 탐색 순서 :
9 1 4 8 0 2 3 5 6 7
>>> |

```

### 03. 과제 내용

- dij  $v_s$   $v_e$ 
  - $v_s$ 로부터 시작해  $v_e$ 로 가는 최단 경로를 다익스트라 알고리즘을 통해 구합니다.

```

C:\Users\WShigGy\sourceWre x + v
그래프 데이터 파일 이름 입력 : graph.txt
그래프 종류 (0: 무방향, 1: 방향) : 1
>>> dij 1 9
Dijkstra 알고리즘 결과 :
Start Node: 1
Path: 1 -> 9
Distance:
    1: 0.0
    2: 2.0
    4: 3.0
    5: 12.0
    6: 7.0
    7: 8.0
    8: 5.0
    9: 2.0

>>> dij 2 8
Dijkstra 알고리즘 결과 :
Start Node: 2
Path: 2 -> 4 -> 8
Distance:
    2: 0.0
    4: 7.0
    5: 10.0
    6: 5.0
    7: 6.0
    8: 9.0
    9: 14.0

>>> |

```

- befo  $v_s$   $v_3$ 
  - $v_s$ 로부터 시작해  $v_e$ 로 가는 최단 경로를 벨만-포드 알고리즘을 통해 구합니다.

```

C:\Users\WShigGy\sourceWre x + v
그래프 데이터 파일 이름 입력 : graph.txt
그래프 종류 (0: 무방향, 1: 방향) : 1
>>> befo 2 3
Bellman-Ford 알고리즘 결과 :
Start Node: 2
Path: No Path
Distance:
    2: 0.0
    4: 7.0
    5: 10.0
    6: 5.0
    7: 6.0
    8: 9.0
    9: 14.0

>>> befo 2 8
Bellman-Ford 알고리즘 결과 :
Start Node: 2
Path: 2 -> 4 -> 8
Distance:
    2: 0.0
    4: 7.0
    5: 10.0
    6: 5.0
    7: 6.0
    8: 9.0
    9: 14.0

>>> |

```

## 03. 과제 내용

- flwa
  - 플로이드-워셜 알고리즘 수행 결과를 출력합니다.

```

C:\Users\WShigGy\source\Wre x + v
그래프 데이터 파일 이름 입력 : graph.txt
그래프 종류 (0: 무방향, 1: 방향) : 1
>>> flwa
Floyd-Warshall 알고리즘 결과 :
Shortest Distance:
INF  4.0  6.0  1.0  4.0  3.0  5.0  6.0  6.0  6.0  INF  INF  INF  INF  INF
INF  INF  2.0  INF  3.0  12.0  7.0  8.0  5.0  2.0  INF  INF  INF  INF  INF
INF  INF  INF  INF  7.0  10.0  5.0  6.0  9.0  14.0  INF  INF  INF  INF  INF
INF  INF  INF  INF  3.0  10.0  12.0  6.0  5.0  10.0  INF  INF  INF  INF  INF
INF  INF  INF  INF  8.0  INF  INF  INF  2.0  7.0  INF  INF  INF  INF  INF
INF  INF  INF  INF  INF  7.0  2.0  3.0  INF  INF  INF  INF  INF  INF  INF
INF  INF  INF  INF  INF  5.0  7.0  1.0  INF  INF  INF  INF  INF  INF  INF
INF  INF  INF  INF  INF  4.0  6.0  7.0  INF  INF  INF  INF  INF  INF  INF
INF  INF  INF  INF  6.0  INF  INF  INF  8.0  5.0  INF  INF  INF  INF  INF
INF  INF  INF  INF  1.0  INF  INF  INF  3.0  8.0  INF  INF  INF  INF  INF
INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF
INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF
INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF
INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF
INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF
>>> |

```



## 03. 과제 내용

- **print**
  - 그래프 정보를 출력합니다.

```

C:\Users\ShigGy\source\repos\DS_HW4_v2\Debug\DS_HW4_v2.exe
그래프 데이터 파일 이름 입력 : graph.txt
그래프 종류 (0: 무방향, 1: 방향) : 0
>>> print
Directed Graph: False
Number of Nodes: 10
0 1 2 3 4 5 6 7 8 9
Adjacency List:
0: (1, 4.0)      (3, 1.0)      (5, 3.0)
1: (0, 4.0)      (2, 2.0)      (9, 2.0)
2: (1, 2.0)      (4, 7.0)      (6, 5.0)
3: (0, 1.0)      (4, 3.0)      (7, 6.0)
4: (2, 7.0)      (3, 3.0)      (8, 2.0)      (9, 1.0)
5: (0, 3.0)      (6, 2.0)      (7, 4.0)
6: (2, 5.0)      (5, 2.0)      (7, 1.0)
7: (3, 6.0)      (5, 4.0)      (6, 1.0)
8: (4, 2.0)      (9, 5.0)
9: (1, 2.0)      (4, 1.0)      (8, 5.0)
Adjacency Matrix:
INF  4.0  INF  1.0  INF  3.0  INF  INF  INF  INF  INF  INF  INF  INF  INF
4.0  INF  2.0  INF  INF  INF  INF  INF  INF  2.0  INF  INF  INF  INF  INF
INF  2.0  INF  INF  INF  7.0  INF  5.0  INF  INF  INF  INF  INF  INF  INF
1.0  INF  INF  INF  INF  3.0  INF  INF  6.0  INF  INF  INF  INF  INF  INF
INF  INF  7.0  3.0  INF  INF  INF  INF  2.0  1.0  INF  INF  INF  INF  INF
3.0  INF  INF  INF  INF  INF  INF  2.0  4.0  INF  INF  INF  INF  INF  INF
INF  INF  5.0  INF  INF  2.0  INF  1.0  INF  INF  INF  INF  INF  INF  INF
INF  INF  INF  6.0  INF  4.0  1.0  INF  INF  INF  INF  INF  INF  INF  INF
INF  INF  INF  INF  INF  2.0  INF  INF  INF  INF  5.0  INF  INF  INF  INF
INF  2.0  INF  INF  1.0  INF  INF  INF  INF  5.0  INF  INF  INF  INF  INF
INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF
INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF
INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF
INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF
INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF  INF
>>>

```

- **exit**
  - 프로그램을 종료합니다.

```

>>> exit

C:\Users\ShigGy\source\repos\DS_HW4_v2\x64\Debug\DS_HW4_v2.exe(프로세스 24252개)이(가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요 ...

```

## 04. 제출물

---

- 아래의 파일을 압축하지 말고 올려주세요.
  1. 구현한 "implementation.h" 파일
    - 채점 자동화 프로그램을 이용해서 채점합니다.
    - 파일 이름은 정확히 "implementation.h"로 해주세요.
    - 파일 이름, 코드 실행 결과 완전히 동일해야 불이익이 없습니다.
  2. 보고서 "hw4\_학번\_이름.pdf" 파일
    - 올려드린 보고서 양식을 기반으로 작성해주세요.
    - 보고서에는 다음의 내용이 포함되어야 합니다.
      - 빈칸 채우기 문제에 대한 정답 (총 10개의 함수)
      - 각 명령어 별 실행 결과
      - 과제를 수행하면서 어려웠던 점 및 극복 과정