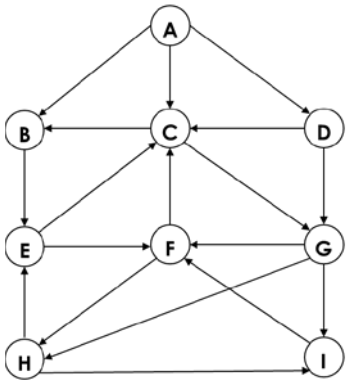
CS3353: Data Structures and Algorithm Analysis I
Spring 2024

Homework #6

- Full name only: _____
- Release date: 4:00 PM, April 17, 2024 (Wednesday)
- Due date: **2:30 PM, May 1, 2024 (Wednesday)**
- It should be done INDIVIDUALLY; Show ALL your work; Submit your source code and results through Canvas.
- Please refer to the course syllabus (Course Requirements and Grading Policy - Assignment, page 3) for the policy of submission, late submission, missing submission, and wrong submission.
- Total: 20 pts

1. Write a program to conduct a depth-first search using a stack and the minimum path search (e.g., breadth-first search) using a queue based on the following directed graph and its adjacency lists. In the depth-first search, any starting node can be selected (e.g., user input) in the graph. In the minimum path search, both starting and ending nodes should be selected (e.g., user input).



Adjacency Lists			
A:	B	C	D
B:	E		
C:	B	G	
D:	C	G	
E:	C	F	
F:	C	H	
G:	F	H	I
H:	E	I	
I:	F		

- Type the homework number and your full name at the top of your source code.

```
/* Homework #6, James Bond */
```

- Your program should be menu-driven and execute the chosen command. If you type 3, then exit the program.

```
                    M E N U

Depth-First Search (0), Minimum Path Search (1)
Exit Program (3)

                    Choose?
```

- Display a message, in the case when searching a path that does not exist in the graph.
- Show ALL your work. For example,

```
                    M E N U

Depth-First Search (0), Minimum Path Search (1)
Exit Program (3)
```



```
                          Choose?  0  H
H  I  F  C  G  B  E


                          M E N U

Depth-First Search (0), Minimum Path Search (1)
Exit Program (3)

                          Choose?  1  A  I

A  C  G  I


                          M E N U

Depth-First Search (0), Minimum Path Search (1)
Exit Program (3)

                          Choose?  3

                    .
                    .
                    .
```

- Please refer to Reference #1 and #2.

- The adjacency list can be implemented by using either a linked list or an array. Both stack and queue data structures should be implemented in your way.

- Submit your source code and self-testing results (e.g., readable and clear screenshots) through Canvas before the due date, **2:30 PM, May 1, 2024 (Wednesday)**. The TA will build and run your source code and test with random input.
  - **Source code (one file only)** – The file name should be "your name + homework number", e.g., james_bond_6.cpp or james_bond_6.java.
  - **Self-testing Results** (e.g., readable and clear screenshots) **in WORD** document

- Grading Policy
  - Compilation error / run-time error: -5 pts
    - Deduct 5 pts first. Then TA needs to evaluate the program and apply the following grading policy.
  - No homework submission: 0 pts
  - Program menu as required: 2 pts
  - Depth-First Search: 7 pts
    - Correct output: 7 pts
    - Incorrect output:
      - The idea/logic of algorithm is wrong: -5
        - Partial points (e.g., 2 pts) are deducted depending on the degrees of wrongness.
  - Minimum Path Search: 7 pts
    - Correct output: 7 pts
    - Incorrect output:
      - The idea/logic of algorithm is wrong: -5

        - Partial points (e.g., 2 pts) are deducted depending on the degrees of wrongness.
    - Self-testing results: 2 pts
        - No testing conducted / No WORD document: -2
    - Use other programming languages other than Java and C++: Deduct 5 pts first. Then TA needs to evaluate the program and apply the above grading policy.
    - Students will not receive 0 pts if students spent time and effort on program and make the submission.
        - If the program has issues/problems, TA needs to evaluate student's program and gives partial points depending on the quality/completion of program.
    - The instructor will decide the grade policy of any scenario which is not covered by the above list. Meanwhile, please kindly contact the instructor if you have any questions regarding the grading policy.

2. **[Extra Credit]** If you can implement the following program, extra 10 points will be provided. Write a program to sort user input data using a set of sorting algorithms.

- Type the homework number and your full name at the top of your source code.

```
/* Homework #6 – Bonus Work, James Bond */
```

- Your program should be a menu-driven and execute the chosen command. If you type 5, then exit the program.

```
                        M E N U

Input Data (0), Insertion Sort (1), Selection Sort (2),
Bubble Sort (3), Shell Sort (4), Exit Program (5)

                       Choose?
```

- A user can input a set of elements and select one of sorting algorithms to sort the recent input set. Show ALL your work. For example,

```
                        M E N U

Input Data (0), Insertion Sort (1), Selection Sort (2),
Bubble Sort (3), Shell Sort (4), Exit Program (5)

                       Choose?  0  9  7  6  15  16  5  10  11

9  7  6  15  16  5  10  11

                        M E N U

Input Data (0), Insertion Sort (1), Selection Sort (2),
Bubble Sort (3), Shell Sort (4), Exit Program (5)

                       Choose?  1

5  6  7  9  10  11  15  16

                        M E N U

Input Data (0), Insertion Sort (1), Selection Sort (2),
Bubble Sort (3), Shell Sort (4), Exit Program (5)
```



```
                        Choose?  0  2  8  6  1  10  15  3  12  11

2  8  6  1  10  15  3  12  11

                         M E N U

Input Data (0), Insertion Sort (1), Selection Sort (2),
Bubble Sort (3), Shell Sort (4), Exit Program (5)

                        Choose?  3

1  2  3  6  8  10  11  12  15
                          .
                          .
                          .
```

- Please refer to the lecture slides for the related sorting algorithms and pseudocodes.

- Submit your source code and self-testing results (e.g., readable and clear screenshots) through Canvas before the due date, **2:30 PM, May 1, 2024 (Wednesday)**. The TA will build and run your source code and test with random input.
  - **Source code (one file only)** – The file name should be "your name + homework number", e.g., james_bond_6_bonus.cpp or james_bond_6_bonus.java.
  - **Self-testing Results** (e.g., readable and clear screenshots) **in WORD** document

- Grading Policy
  - Compilation error / run-time error: -5 pts
    - Deduct 5 pts first. Then TA needs to evaluate the program and apply the following grading policy.
  - No homework submission: 0 pts
  - Program menu as required: 0.5 pt
  - Insertion Sort: 2 pts
    - Correct algorithm implementation and output: 1 pt
    - Incorrect algorithm implementation and output:
      - The idea/logic of algorithm is wrong: -1
        - Partial points (e.g., 0.5 pt) are deducted depending on the degrees of wrongness.
  - Selection Sort: 2 pts
    - Correct algorithm implementation and output: 1 pt
    - Incorrect algorithm implementation and output:
      - The idea/logic of algorithm is wrong: -1
        - Partial points (e.g., 0.5 pt) are deducted depending on the degrees of wrongness.
  - Bubble Sort: 2 pts
    - Correct algorithm implementation and output: 1 pt
    - Incorrect algorithm implementation and output:
      - The idea/logic of algorithm is wrong: -1
        - Partial points (e.g., 0.5 pt) are deducted depending on the degrees of wrongness.
  - Shell Sort: 2 pts
    - Correct algorithm implementation and output: 1 pt
    - Incorrect algorithm implementation and output:
      - The idea/logic of algorithm is wrong: -1

        - Partial points (e.g., 0.5 pt) are deducted depending on the degrees of wrongness.
- Self-testing results: 0.5 pt
    - No testing conducted / No WORD document: -0.5
- Use other programming languages other than Java and C++: Deduct 5 pts first. Then TA needs to evaluate the program and apply the above grading policy.
- Students will not receive 0 pts if students spent time and effort on program and make the submission.
    - If the program has issues/problems, TA needs to evaluate student's program and gives partial points depending on the quality/completion of program.
- The instructor will decide the grade policy of any scenario which is not covered by the above list. Meanwhile, please kindly contact the instructor if you have any questions regarding the grading policy.

