1. (8%) Questions 1.1 to 1.8. For each statement, choose (a) if there is no error, (b) if there is a compiler error, or (c) for undefined behavior (in this case, program might crash). One point (1%) for each.

1.1 ( ) int *p = 0;　　(a) no error (b) compile error (c) undefined behavior
1.2 ( ) cout << *p << endl;　　(a) no error (b) compile error (c) undefined behavior
1.3 ( ) p = new int [10];　　(a) no error (b) compile error (c) undefined behavior
1.4 ( ) int *q = p + 3;　　(a) no error (b) compile error (c) undefined behavior
1.5 ( ) cout << p – q << endl;　　(a) no error (b) compile error (c) undefined behavior
1.6 ( ) p[10] = 0;　　(a) no error (b) compile error (c) undefined behavior
1.7 ( ) q = 4;　　(a) no error (b) compile error (c) undefined behavior
1.8 ( ) delete p;　　(a) no error (b) compile error (c) undefined behavior

2. (2%) Questions 2.1 to 2.2. Choose the best answer. One point (1%) for each.

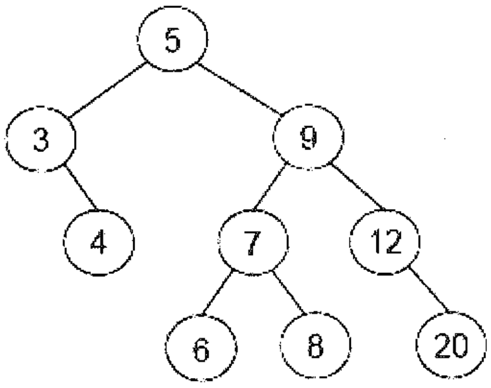
2.1 ( ) Which of the following statements about classes is true?
(a) A class can contain both data members and methods　(b) The members of a class may be private or public
(c) The name of the constructor is always the same as the name of the class
(d) The class definition contains prototype statements for the methods
(e) All of the statements (a-d) are true　(f) none of the above

2.2 ( ) The null terminator is _____.
(a) used to advance to a new line when executing a **cout** statement　(b) represented as '\0'
(c) used to end a valid string　(d) All of the statements (a-c) are correct
(e) Only statements b and c are correct　(f) none of the above

3. (15%) Questions 3.1 to 3.15. Circle **T** for true or **F** for false. One point (1%) for each.

3.1 ( ) A derived class may access inherited private data members.
3.2 ( ) A derived class constructor should initialize inherited data members.
3.3 ( ) You cannot override a function unless it is virtual.
3.4 ( ) The default behavior of the assignment operator in a derived class is to call the assignment operator of the base class.
3.5 ( ) A base class with an empty virtual destructor should define an empty virtual copy constructor and empty virtual assignment operator.
3.6 ( ) When overriding a member function, it should take the same parameters.
3.7 ( ) A pointer to base class may point to a derived object.
3.8 ( ) Derived constructors override base constructors.
3.9 ( ) Making a function virtual increases the size of every instance.
3.10 ( ) A class with pure virtual functions is abstract.
3.11 ( ) If B is an abstract base class, you may have objects of type B.
3.12 ( ) If B is abstract, then a function may have parameters of type B&
3.13 ( ) If B is abstract, then you may have a pointer of type B*
3.14 ( ) If class A is declared as a member of class B, then B is a base class.
3.15 ( ) If class A is declared as a member of class B, then A may access the protected members of class B.

4. (2%) The binary search tree shown below was constructed by inserting a sequence of data items into an empty tree. Which of the following input sequences *cannot* produce this binary search tree?



(a) 5, 3, 4, 9, 12, 7, 8, 6, 20
(b) 5, 9, 3, 7, 6, 8, 4, 12, 20
(c) 5, 9, 7, 8, 6, 12, 20, 3, 4
(d) 5, 9, 7, 3, 8, 12, 6, 4, 20
(e) 5, 9, 3, 6, 7, 8, 4, 12, 20

5. (3%) An array with a sequence of numbers {50, 35, 40, 25, 20, 33, 27} is organized into a max-heap. Which array represents the heap after two deleteMax operations are performed?

(a)

35	27	33	25	20

(b)

40	25	20	33	27

(c)

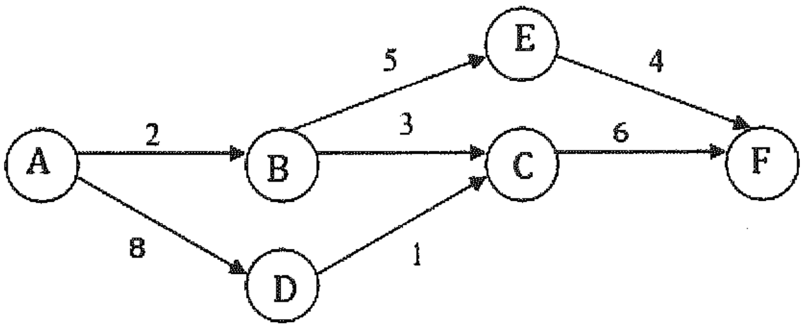
35	25	20	33	27

(d)

35	33	27	20	25

(e) None of the above

6. (6%) Given a directed graph, please answer the following questions.

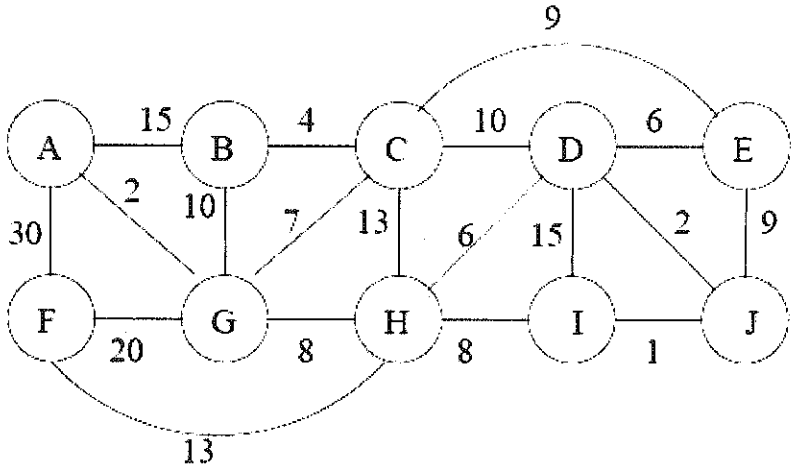


(a) (2%) Perform Dijkstra's algorithm to determine the shortest path from vertex A to each of the other vertices. Please list the vertices in the order they are processed by the algorithm.

(b) (2%) Perform a depth-first traversal of the graph, starting with vertex A. Select the smallest edge first, when appropriate.

(c) (2%) What is a topological ordering of the above graph? You must explain the approach.

7. (3%) Please use Kruskal's algorithm to find the Minimum Spanning tree for the following graph. Provide the edges in the order in which they would be found by Kruskal's algorithm.



8. (2%) What is the postfix expression of ((2+2)*3)*((3-1)*2)?

9. (9%) To build a binary search tree (BST), the following numbers are inserted in the given order: {50, 80, 20, 100, 60, 95, 52, 90, 62, 70, 120, 66, 85, 68, 87, 64}.

(a) (4%) Please draw the binary search tree.

(b) (5%) Delete element 80 in the BST built in (a). Mark the operations on the given BST and briefly explain your procedure. Then, redraw the final BST after the deletion.

10. (6%) (True or false. If the statement is false, correct the wrong part. Simply negate the statement is not accepted. 2% each)

(a) If a problem X can be reduced to a known NP-hard problem, then X must be NP-hard.

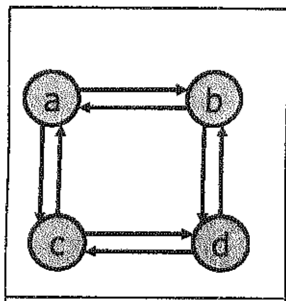
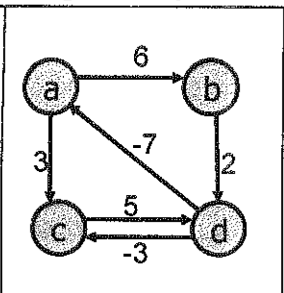
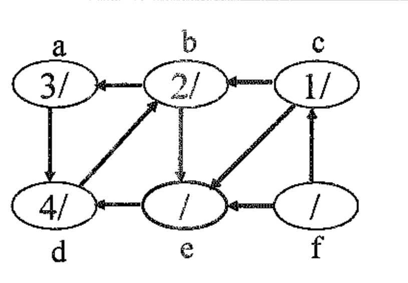
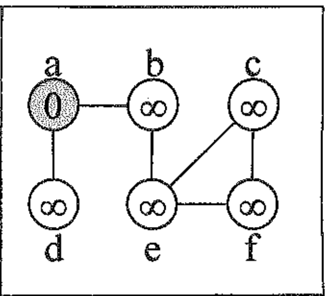
(b) The solution of $T(n) = 8T(n/2) + \theta(n^2\sqrt{n})$ is $\theta(n^3)$.

(c) Using depth first search to explore a graph will also give you the shortest path from the starting node.

11. (3%) Given the graph in Fig. 11 and try to find a simple path from node a to node d with the most edges. Does such problem have optimal substructure? Prove, or disprove with a counter example.

12. (6%) (a) (4%) Convert the negative edges of the graph in Fig. 12 to nonnegative ones without changing the properties of the original shortest paths; (b) (2%) If a, b, c, d represent clusters of vertices and edges, which approach is more efficient to detect whether there is a negative weight cycle or not when the whole graph is sparse?

13. (10%) (a) (4%) Continue the graph search in Fig. 13.1 and Fig. 13.2, and draw the resulted trees; (b) (2%) What are the names of the approaches? (c) (4%) Perform topological sort on both Fig. 13.1 and Fig. 13.2. Write down their results if doable. If not, explain why.

		 a b c 3/ 2/ 1/ 4/ / / d e f	
Fig. 11	Fig. 12	Fig. 13.1	Fig. 13.2

14. (9%) Show the output of the following program.

```
#include <stdio.h>

void reverse(char s[], int n) {
  char c;
  int i;
  for (i=1; i<n/2; i++) {
    c = s[i];
    s[i] = s[n-i-1];
    s[n-i-1] = c;
  }
}

main() {
  char A[9];
  int i;
  int *p;
  p = &A[2];
  for (i=0; i<9; i++) { A[i] = 0; }
  A[0] = 'a';
  printf("%s\n", A);
  for (i=1; i<5; i++) { A[i] = A[i-1] + i; }
  printf("%s\n", A);
  reverse(p, 5);
  printf("%s\n", p);
}
```

15. (8%) What will be the output? Assume the program is named **prog**

prog 1 2 3 4 5

```
#include <stdio.h>
main(int argc, char **argv) {
   char text[] = "sweetapplepie";
   for (i=0; i<argc; i++) { strcat(text, argv[argc-i-1]); }
   len = strlen(text);
   text[len<<1] = '\0';
   printf("%s\t%d\t%s\n", argv[0], argc, text);
}
```

16. (8%) Please write a function swap() such that the output of the following program will be 300, 500. You need to fill the parameters in the function call too.

```
#include  <stdio.h>
void swap(     ,     ) {

}

main() {
   int A, B;
   A = 500;
   B = 300;
   swap(__, __); //
   printf("%d, %d\n", A, B);
}
```