Section A

Total marks: 40

Answer **ALL** questions in this section.

1. Computer memory leak simply refers to

   A. unused memory space.

   B. memory space not accessible for use due to bad program logic.

   C. not enough memory spaces in a computer system.

   D. unavailable memory spaces due to overuse.

2. Why is `x == &x[0]` always **True** if `x` is a pointer variable?

   A. `x` always stores the memory location of `x[0]`.

   B. `x` always stores the value of `x[0]`.

   C. `x` always stores the memory location of the last entry.

   D. `x` always stores the value of the last entry.

3. Header nodes are important for linked list implementation because

   A. insertions into a linked list must follow some existing item.

   B. removal of the first item in the linked list is never allowed.

   C. whenever an item `x` is removed the compiler automatically cleans it up.

   D. insertions allow the placement of an item in the first position of the linked list.

4. The rule $T1(N) * T2(N) = O(f(N) * g(N))$ simply means

   A. the growth rates of two different multiplying functions can be multiplied together.

   B. the orders of two different multiplying functions can be summed up together.

   C. the efficiencies of two different algorithms equals their multiplied output.

   D. the orders of two different algorithms can be found by amplifying them up.

5. Which of the following does `C++` allow for freeing memory spaces?

   A. delete B. remove C. unreference D. current->next;

6. The growth rate of the function $f(n) = 2^{2+n}$ is

   A. 2^n B. $2n$ C. 2^{2n} D. 2^{2+n}

7. N^3 grows at the same rate as which of the following?

   A. $2N^3$ B. 3^N C. $3N^2$ D. N^{3N}

8. If for $T(N) = o(p(N))$, for all positive constants c, there exists an n_0 such that $T(N) < c \cdot p(N)$ when $N > n_0$, then that defines

   A. Small-Oh B. Big-Oh C. Big-Theta D. Big-Omega

9. Which algorithm analysis methodology assumes equal order of growth among two different algorithms?

    A. Big Theta analysis

    B. Big Oh analysis

    C. Small-Oh analysis

    D. Big-Omega analysis

10. If for $T(N) = O(f(N))$ there are positive constants c and n_0 such that $T(N) \leq c \cdot f(N)$ when $N \geq n_0$ is the mathematical definition for

    A. Big Oh Analysis

    B. Big-Omega Analysis

    C. Big-Theta Analysis

    D. Small-Oh Analysis

11. Algorithm analysis is basically concerned with

    A. code benchmark analysis

    B. computing resources

    C. readability

    D. all mentioned

12. In the function definition below

```
void printInfo(const Student & s)
```

    A. `printInfo` does not return any value.

    B. the parameter, s is passed by reference.

    C. the parameter is a read-only parameter.

    D. All given descriptions are true.

13. The exact value of the function depends on many factors, including

    A. The speed of the host machine.

    B. The quality of the compiler.

    C. The quality of the program.

    D. All mentioned

14. Which of the following is **not** a computing resource in algorithm analysis?

    A. Running time   B. Execution time   C. Computing memory   D. None mentioned

15. A step-by-step procedure for performing some task in a finite amount of time that has been encoded into some programming language is a ________.

    A. data structure   B. program   C. algorithm   D. experimental analysis

Please go on to the next page...

9. Which algorithm analysis methodology assumes equal order of growth among two different algorithms?
   A. Big-Theta analysis
   B. Big-Oh analysis
   C. Small-Oh analysis
   D. Big-Omega analysis

10. If for $T(N) = O(f(N))$ there are positive constants c and n_0 such that $T(N) \leq c \cdot f(N)$ when $N \geq n_0$ is the mathematical definition for
    A. Big-Oh Analysis
    B. Big-Omega Analysis
    C. Big-Theta Analysis
    D. Small-Oh Analysis

11. Algorithm analysis is basically concerned with
    A. code benchmark analysis
    B. computing resources
    C. readability
    D. all mentioned

12. In the function definition below

```
void printInfo(const Student & s)
```

    A. `printInfo` does not return any value.
    B. the parameter, `s` is passed by reference.
    C. the parameter is a read-only parameter.
    D. All given descriptions are true.

13. The exact value of the function depends on many factors, including
    A. The speed of the host machine.
    B. The quality of the compiler.
    C. The quality of the program.
    D. All mentioned

14. Which of the following is **not** a computing resource in algorithm analysis?
    A. Running time B. Execution time C. Computing memory D. None mentioned

15. A step-by-step procedure for performing some task in a finite amount of time that has been encoded into some programming language is a ________.
    A. data structure B. program C. algorithm D. experimental analysis

Please go on to the next page...

16. The efficiency for the following piece of code

```
for ( i = 0 ; i < n ; ++i )
  for ( j = 0 ; j < n ; ++j )
    ++k ;
```

is A. $O(N^2)$ B. $O(N)$ C. $O(2^N)$ D. $O(2N)$

17. In order to use the *manipulator* `setprecision()` in a `C++` program which of the following header is needed?
A. `<iomanip>` B. `<iostream>` C. `<ctime>` D. `<cprecision>`

18. The following piece of `C++` code

```
current->next = new Node( x, current->next );
```

A. creates a new node.

B. places element `x` in a node.

C. points a node to another node.

D. does all the statements stated above.

19. A fundamental advantage of a linked list over traditional arrays is that

A. changes to a linked list can be made by using only a constant number of data movements.

B. changes to a linked list can be made in logarithmic times.

C. linked list are usually easier to set up and implement.

D. arrays are much easier to set up, implement and maintain.

Answer **True** (T) or **False** (F) for questions **20 to 40** in the spaces provided.

20. In the `C++` expression `*x + 5`, the value 5 is added to the *dereferenced* value of `x`. ___

21. In `C++`, primitive arrays cannot be copied or compared because the array name is merely an address. ___

22. If `ptr` is uninitialized in a computer program, the assignment `*ptr = x` is likely to cause problems. ___

23. Given that a certain pointer `ptr` points to a structure, `s`, then `(*ptr).maxMark` points the pointer to the member `maxMark` of `s`. ___

24. If `gradePoint` is a member of a structure, `s`, then in `(*ptr).gradePoint` the parentheses are necessary because of pointer arithmetic. ___

25. An that requires several bytes of main memory is not useful even if it is completely correct. ___

26. In a linked list each Node contains two fields one of which is a <strong>reference</strong> to itself. ___

27. If N^3 grows faster than N^2, we can say that $N^3 = O(N^2)$. ___

28. $f(N) = N^2$ grows at the same rate as $g(N) = 2N^2$. ___

29. If $g(N) = 2N^2$ then we can also say that $g(N) = O(N^4)$. ___

30. Given that a certain pointer, `ptr`, points to a structure, `s`, then `ptr->maxMark` accesses the member `maxMark` of `s`. ___

31. The greatest shortfall of dynamically expanding arrays is simply that we can easily run out of memory for a running program. ___

32. A downside of linked-list is that an arbitrary item can no longer be found in one access. ___

33. The order of any log-squared function is always also *constant*. ___

34. Logarithmic algorithm usually grow very slowly. ___

35. Exponential algorithms are usually have the highest growth rates. ___

36. $f(N) \geq O(g(N))$ essentially means nothing and makes no sense. ___

37. $N^2 log\ N$ and $N log\ N$ essentially have the same growth rate. ___

38. Arrays are not suitable for list implementation if deletions can occur at the end of the list. ___

39. A linked can be implemented without the use of header nodes. ___

40. Linked-lists are answers to dynamically expanding arrays in contiguous memory space. ___

Section B

Total marks: 60

Answer any **TWO** (2) questions in this section.

Question 1: Algorithm Analysis, Order of Magnitude Notation . . . . *30 marks*

(a) **[3 marks]** Why is *experimental analysis* of algorithms (i.e. using running times of different algorithms) not a good measure of efficiency of algorithms? Give three reasons.

(b) **[2 marks]** In analysing algorithms what do we generally mean by *computing resources*? State briefly.

i. **[2 marks]** An algorithm running a *linear-time* efficiency may not necessarily be chosen over another running a *log-time* because of some important considerations. What are some of these?

(c) **[2 marks]** At the design *stage* of an algorithm to solve a problem why is the use of experimental analysis not recommended? State briefly.

(d) **[2 marks]** Why is it important to use relative growth rates to measure algorithms?

(e) We wish to analyse the performance of the following algorithm using **Big 'O' Notation** to determine the execution time function, $T(n)$ defined below. Use the code to answer the questions that follow.

```
1     int a = 5
2     int b = 6
3     int c  = 10
4
5     for(int i = 0; i < n; i++){
6       for(int j = 0; j < m; j++){
7          int x = i * j
8          int y = j * j
9          int z = i * j
10      }
11    }
12
13    for(k = 0; k < n; k++){
14      int w = a * k + 45
15      int v = b * b
16    }
17
18    int b = 33
```

i. **[3 marks]** What do lines 1 to 3 represent in the function $T(n)$?

ii. **[3 marks]** What do lines 7 to 9 represent in the function $T(n)$?

iii. **[2 marks]** What do lines 14 & 15 represent in the function $T(n)$?

Question 1 continues. . .

iv. [1 mark] What does line 18 represent in function $T(n)$?

v. [2 marks] What is the final function $T(n)$ of this piece of code?

vi. [2 marks] What will be the performance of $T(n)$ in Big 'O' Notation?

(f) Find the running times of the following functions using Big 'O' Notation:

i. [2 marks] $5n^2 + 3nlog\ n + 2n + 5$

ii. [2 marks] 2^{n+2}

(g) [2 marks] Order the following functions by growth rate: $Nlog^2N, NlogN, 2^N/2, N$.

Question 2. Linear Data Structures, Stacks, Queues, Linked Lists, Trees . . . . . *30 marks*

(a) [**2 marks**] What, generally speaking, do you think distinguishes one *data structure* from another?

(b) [**2 marks**] Briefly explain what *singly linked list* is in your own words.

(c) The following is a pictorial representation of how an insertion may be done in linked lists. This is followed by the actual computer code that makes this possible. Answer the questions that follow below:

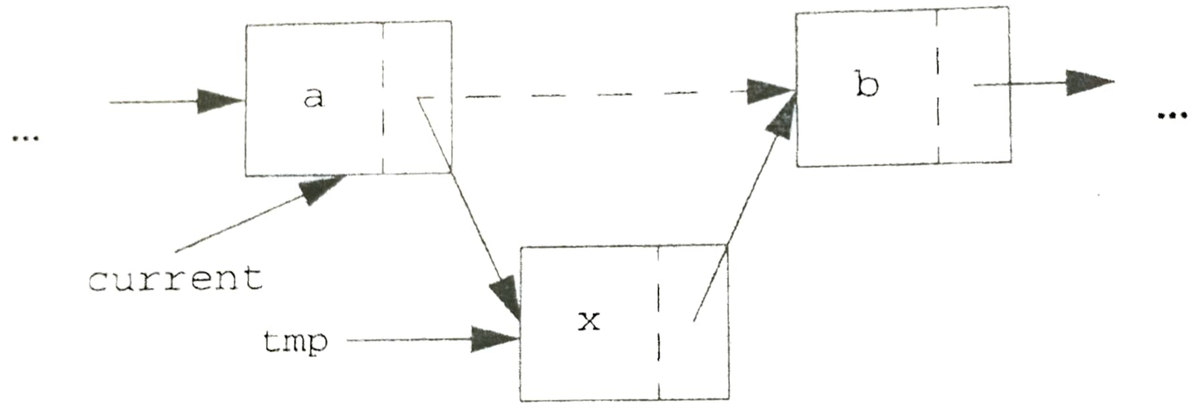


Figure 1. Making an insertion into a linked list.

```
1    tmp = New Node;
2    tmp->element = x;
3    tmp->next = current->next;
4    current->next = tmp;
```

i. [**2 marks**] What does **line 1** of the code do?

ii. [**2 marks**] What does **line 2** of the code do?

iii. [**2 marks**] What does **line 3** of the code do?

iv. [**2 marks**] What does **line 4** of the code do?

v. [**2 marks**] If the **Node** has a constructor that initializes the data members directly how can be code be simplified?

vi. [**2 marks**] What is a *linked list* ADT?

Question 2 continues. . .

(d) [2 marks] What does **FIFO** mean in abstract data types? How does this describe the *queue*?

(e) [2 marks] Give two practical applications of a *queue* ADT.

(f) [2 marks] What are the defining characteristics of the *queue* ADT?

(g) [2 marks] Give two practical applications of a *stack* ADT.

(h) [2 marks] A linked-list abstraction does not necessarily require a contiguous space in memory. Explain.

   i. [2 marks] A node in a *linked-list* object must include at least *two* pieces of vital information. What are they?

   ii. [2 marks] What is the *only* entry point for a *linked-list* ADT?

Question 3: Trees, Binary Trees, Hashing & Hash Functions. . . . . *30 marks*

(a) [2 marks] Give the reason why trees are important in computer science.

(b) Use the tree below to answer the questions that follow.

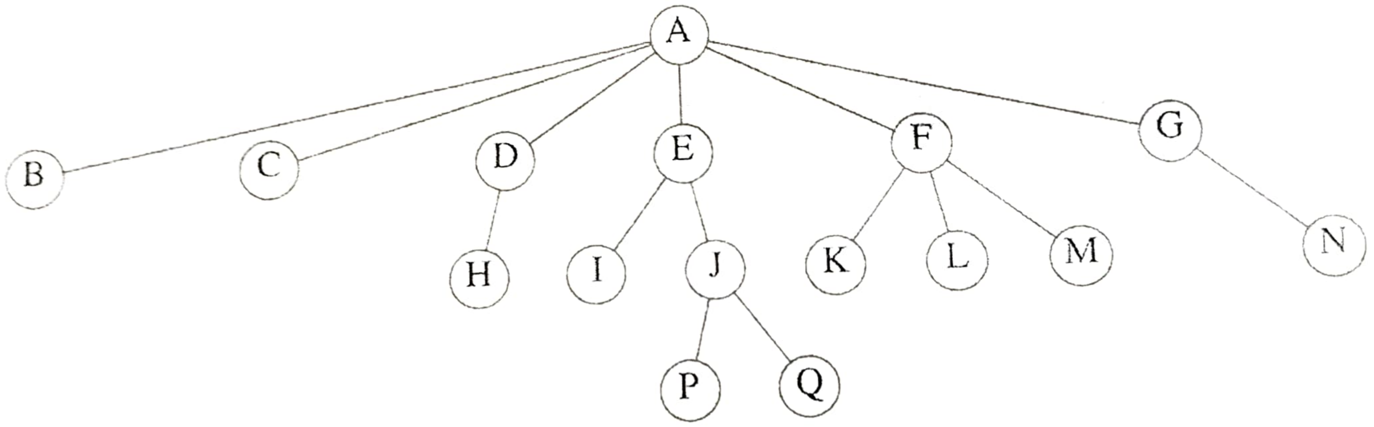


Figure 1: A tree

   i. [2 marks] In Figure 1 what is the relationship between K, L and M to F and A?

   ii. [5 marks] What are the *leaves* in this tree of Figure 1?

(c) [2 marks] Define the *path* of a tree.

(d) [2 marks] Define the *depth* of a tree.

   i. [1 mark] What is the depth of A in Figure 1 above?

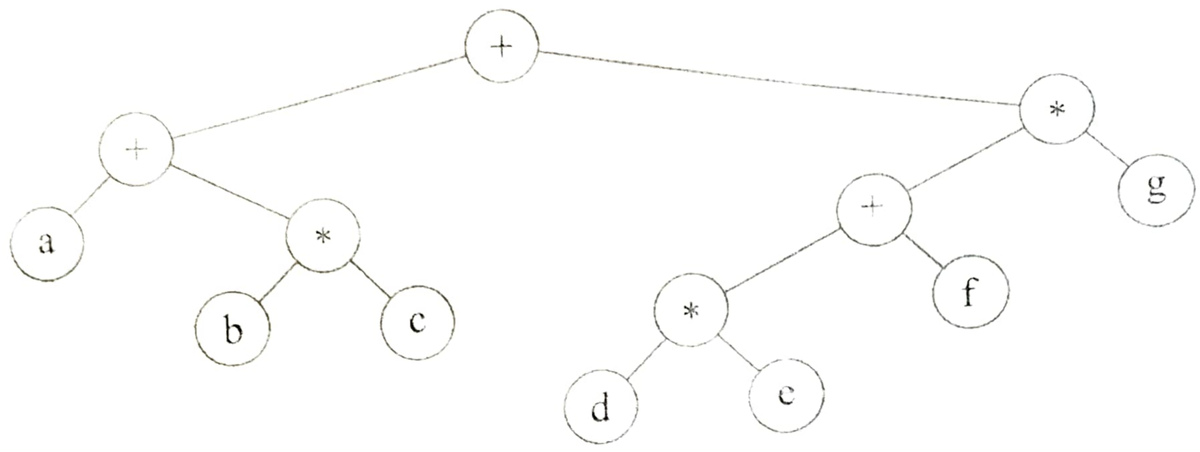
(e) [2 marks] Define the *height* of a tree.

   i. [1 mark] What is the height of Q in Figure 1 above?

(f) [2 marks] What is a **binary tree**?

(g) [3 marks] What expression does the following *expression tree* resolve to? (*Do not show how you arrived at your answer.*)

Question 3 continues. . .



(h) [5 marks] Assume that we have the set of integer items 54, 26, 93, 17, 77, and 31. Using the *remainder* method fill up a **hash table** with a table size of 11 with these integers.

(i) [3 marks] What is the **load factor**, λ, for the hash table in **h** above?

Question 4: Arrays, Pointers, Pointer Arithmetic .... *30 marks*

(a) [2 marks] What is a *pointer*?

(b) [3 marks] Why is **line 1** of the following piece of C++ code *illegal*?

```
1    int *ptr = & x;
2    int x = 5;
3    int y = 7;
```

(c) [2 marks] Define an **aggregate** as a concept in C++ programming.

i. [2 marks] Give two examples of an aggregate in C++.

(d) [2 marks] What is the use of the **array indexing operator** [ ]?

(e) [2 marks] What is the use of the **dot** operator?

(f) [5 marks] Match up the following C++ expressions (Code & Meaning) if x is a pointer object (*Draw a new table with the right pairs*):

Code	Meaning
1. `5 + x[ 0 ]`	A. Always true.
2. `0 == x[ 0 ]`	B. Same as `++*x`
3. `++x[ 0 ]`	C. True if `x[0]` is 0
4. `x++[ 0 ]`	D. Same as `*x++`
5. `x == &x[ 0 ]`	E. Add `[0]` and 5

(g) Consider the following piece of C++ code.

```
#include <iostream>
using namespace std;

```

Question 4 continues...

```
 4  #define MAXLEN 80
 5
 6  int main()
 7  {
 8    char line[MAXLEN], *p;
 9
10    cout << ‘‘Enter a line of text: ’’ << endl;
11
12    // Input a line:
13    for( p = line;
14         p < line + MAXLEN && cin.get(*p) && *p != ‘\n’;
15         ++p )
16    ;
17
18    while( --p >= line )
19        cout << *p;
20
21    cout << endl;
22
23    return 0;
24  }
```

i. [**2 marks**] What does **line 8** of the code achieve?

ii. [**10 marks**] What should be the output of this program if the string “`Hello World`” is given as input?

End of exam