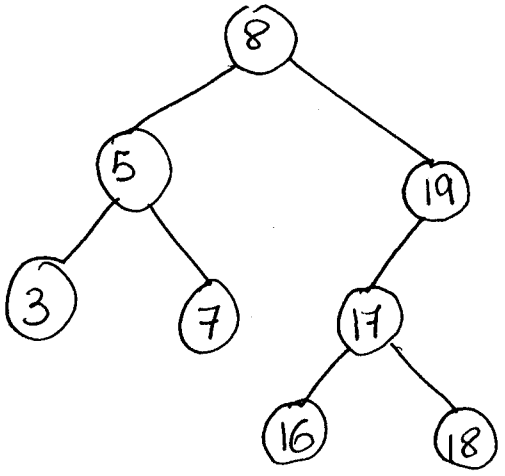
Problem 1 (10 points)

Part a. What is an ordered binary tree? (3 points)

An ordered rooted tree is a rooted tree where the children of each internal vertex are ordered. In an ordered binary tree, the two possible children of a vertex are called the left child and the right child, if they exist. In addition, they retain the property :

left_child < root < right_child

Part b. Given the following inputs 8, 5, 3, 7, 19, 17,16,18. Diagrammatically show the ordered binary tree with 8 as the root node. (4 points)



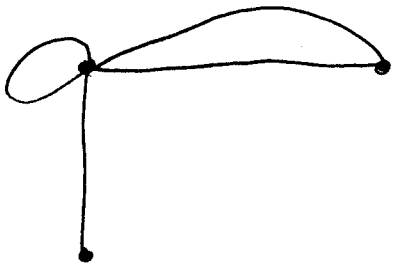
Part c. Is the tree you drew balanced? Justify your answer. (3 points)

A ordered tree is said to be balanced if all the leaf nodes are in level h or h-1 where h is the height of the tree. ∴ THE TREE IS BALANCED.

Problem 2. (5 points)

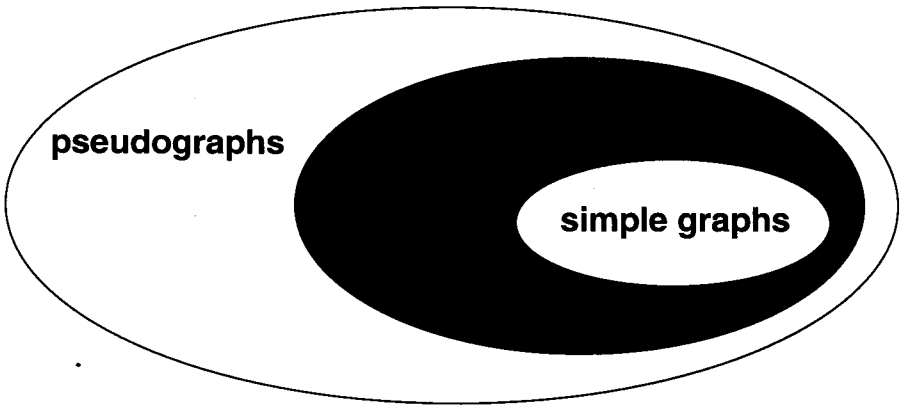
Part a. What is a pseudograph? Draw a simple pseudograph based on your definition. (3 points)

A pseudograph is a graph in which multiple edges are allowed between the same pair of vertices. In addition, a vertex may connect to itself via an edge (i.e. loops are allowed).



Part b. Are all graphs pseudographs? Justify your answer. (2 points)

The set of pseudographs subsumes all other kinds of graphs. Remember the definition allows multiple edges between the same pair of vertices. It does not state that there has to be two or more edges connecting the same pair of vertices.



3. (15 points)

Part a. Write Pseudo code to perform insertion-sort on an array containing 10 integers. (5 points)

Preconditions: Integer array of size 10.
Postconditions: Sorted Array of Integers

Pseudocode:

- For J in 2 .. 10 loop
  - Key := A(J)
  - Index := J-1
  - While I > 0 and A(I) > Key
    - Shift elements one space to the right
    - Decrement I
  - Copy the value of key into A(I+1)

Part b. Implement your algorithm in Ada95. (8 points)

```
Integer_Array_Size: constant Integer :=10; --define the array size to be be 10

type Integer_Array is array (1.. Integer_Array_Size) of Integer;

procedure Sort_Array (A : in out Integer_Array ) is

  i,j,key : integer;

  begin
    for J in 2 .. Integer_Array_Size loop
      I := J-1;
      key := A(J);
      while (I > 0) loop
        if (A(I) > Key) then
                        A(I+1) := A(I);
              I := I -1;
        else
              exit;
        end if;

      end loop;
       A(I+1):= key;
    end loop;
    for I in 1 .. 10 loop
      Put(A(I));
      New_Line;
    end loop;

end;
```

Part c. Assume the input is (2 points)

10	-3	32	23	1	5	32	4	8	0

Show the contents of the sorted array.

-3	0	1	4	5	8	10	23	32	32

Problem 4. (50 points)
Given the structure of the node as follows:

Name	Id	Next

Where:

Name: is a string of 80 characters
Id: is an integer
Next: is a pointer to a node

Part a. Write pseudo-code to sort a linked list based on the id. (20 points)
Assume that the head of the list is given to you.

Preconditions: Head pointer of the list
Postcondtions: Head pointer points to the sorted list.
Pseudocode :

- Create a new head pointer, say new_list
- Set New_List to null
- Create a temporay pointer called current_node
- Set Current_node to Head
- While current_node /= null do
    - Next_node := current_node.next;
    - Current_node.next := null
    - Call Insert_In_Order with New_List and Current_Node
    - Reset Current_node to Next_Node
- Set Head to New_List

- Part b.

i. What is the Ada95 declaration of the record used to represent the node? (5 points)

```
type Node;
type NodePtr is Access Node;
type Node is record
        Name: String (1..80)
        Id : Integer;
        Next : NodePtr;
end record;
```

ii. Complete the Ada95 procedure (shown below) to implement your sorting algorithm. (15 points)

```
procedure Sort ( Head : in out NodePtr) is

  procedure Insert_In_Order (
    New_List    : in out NodePtr;
    Insert_Node : in     NodePtr) is
   Prev, Temp : NodePtr;
  begin
   if New_List = null then
    New_List := Insert_Node;
   else
    Temp:= New_List;
    while Temp/= null loop
     if (Temp.Id < Insert_Node.id) then
      Prev := Temp;
      Temp := Temp.Next;
     else
      exit;
     end if;
    end loop;

    if Temp = null then
```

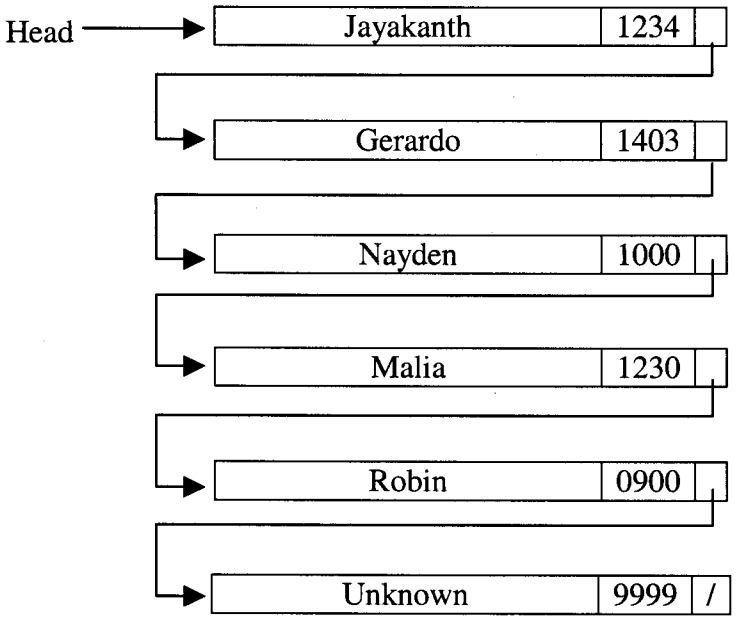
```
          Prev.Next := Insert_Node;
        else
          if Prev = null then
            New_List := Insert_Node;
          else
            Prev.Next := Insert_Node;
          end if;
        end if;
        Insert_Node.Next := Temp;
      end if;
    end Insert_In_Order;


    Current_Node,
    Next_Node    : NodePtr;
    New_List     : NodePtr;

  begin
    New_List := null;
    if Head /= null then
      Current_Node:= Head;
      while Current_Node /= null loop
        Next_Node:= Current_Node.Next;
        Current_Node.Next := null;
        Insert_In_Order(New_List, Current_Node);
        Current_Node:= Next_Node;
      end loop;
      Head := New_List;
    end if;
  end Sort;
```

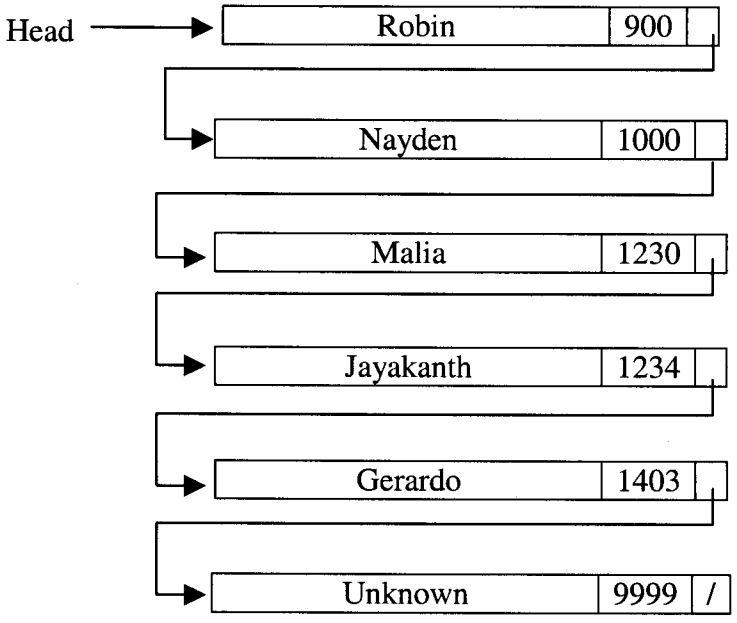
Part c. What is the Time complexity (Big O) of your algorithm? (5 points)

The algorithm has a time complexity of $O(n^2)$ because

- The insert_in_order procedure takes O(n)
- And the while loop executes atmost n times.

Part d. Show the result of applying your algorithm to the list below. (5 points)





Problem 5. (10 points)

Convert the following infix expression into:

((a + b) * c / d –e + f)/(g + (h – i) * j)

Part a. Equivalent Prefix expression (5 points)

/ - * + a b / c d + e f + g * - h i j

Part b. Equivalent Postfix expression (5 points)

a b + c * d / e – f + g h i – j * + /

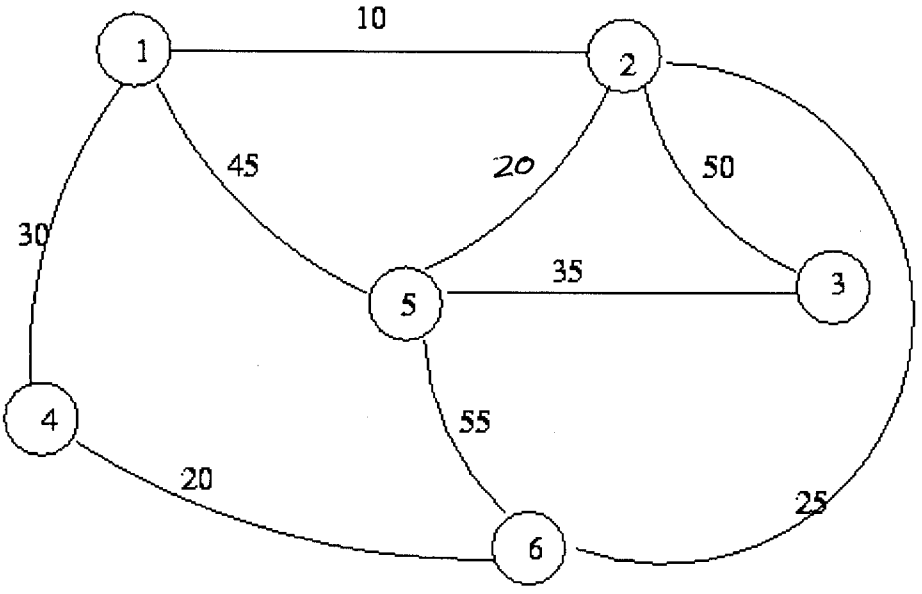
Problem 6. (15 points)

Part a. What is a Minimum Weight Spanning Tree(MST)? (5 points)

Given a graph G(V,E), a minimum weight spanning tree T(V', E') is a subgraph of G with the following properties

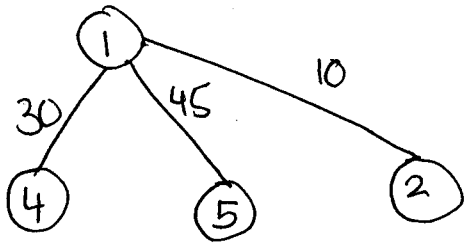
- T is a tree
- V' = V
- The sum of E' is the lowest for all possible spanning trees of G (V,E)

Part b. Given the following graph, find the minimum weight spanning tree using Prim-Dijkstraa's algorithm. Show graphically the steps taken for obtaining the MST. (10 points)

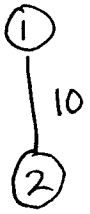


STARTING AT NODE 1

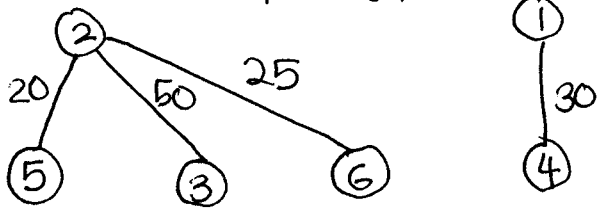
FRINGE_SET



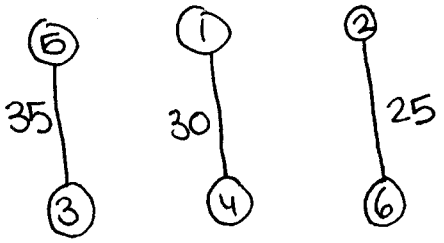
MST



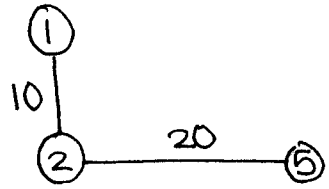
FRINGE_SET



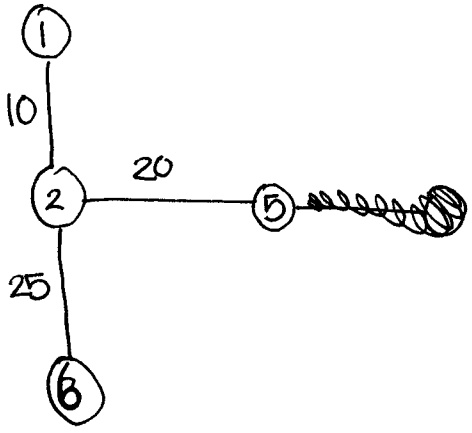
FRINGE_SET

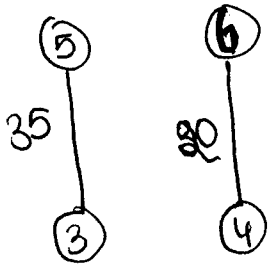


MST

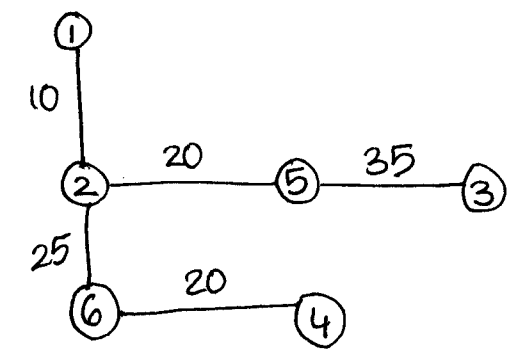


MST





MST



weight of the MST

= 10 + 20 + 25 + 35 + 20

= 110

Problem 7. (15 points)
Part a. What are the three factors influencing the performance of hashing? Explain each of them in 5 sentences. (9 points)

There are three factors the influence the performance of hashing:

- Hash function
  - should distribute the keys and entries evenly throughout the entire table
  - should minimize collisions

- Collision resolution strategy
  - Open Addressing: store the key/entry in a different position
  - Separate Chaining: chain several keys/entries in the same position

- Table size
  - Too large a table, will cause a wastage of memory
  - Too small a table will cause increased collisions and eventually force *rehashing* (creating a new hash table of larger size and copying the contents of the current hash table into it)

The size should be appropriate to the hash function used and should typically be a prime number.

Part b. What are the three techniques used to compute the hash function. Give an example for each of them. (6 points)

- Modular Arithmetic: Compute the index by dividing the key with some value and use the remainder as the index. This forms the basis of the next two techniques.

  For Example: index := key MOD table_size

- Truncation: Ignoring part of the key and using the rest as the array index. The problem with this approach is that there may not always be an even distribution throughout the table.

  For Example: If student id's are the key 928324312 then select just the last three digits as the index i.e. 312 as the index. => the table size has to be atleast 999. Why?

- Folding: Partition the key into several pieces and then combine it in some convenient way.

  For Example:
  - For an 8 bit integer, compute the index as follows:
    Index := (Key/10000 + Key MOD 10000) MOD Tab

Problem 8 (10 points)
Part a. The use of simple parity allows detection of single bit errors. Explain why the statement is true. (2 points)

Codes with a parity bit of one, have a hamming distance of 2. Hence we can detect 1 error.

Part b. A source generates letters from the alphabet {a1, a2, a3, a4} with corresponding probabilities {1/2, 1/4, 1/8, 1/8}.

i. What is the binary entropy of the source? (3 points)

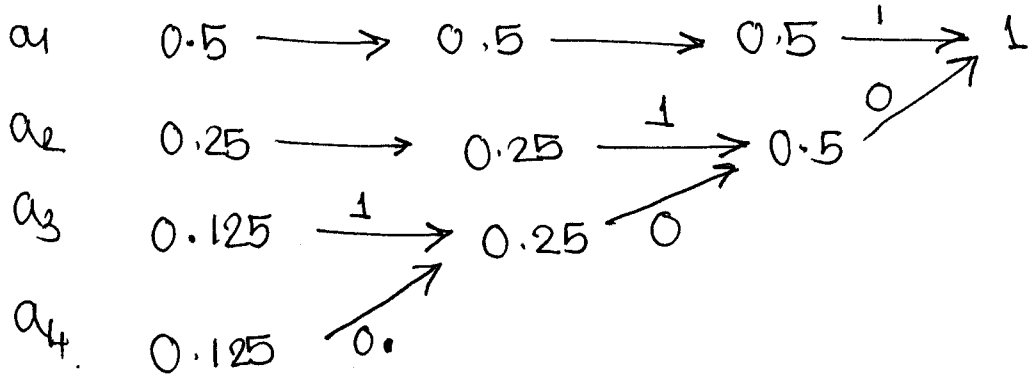
$$\sum_{i=1}^{4} p \log \frac{1}{p}$$

$$= \frac{1}{2}\log 2 + \frac{1}{4}\log 4 + \frac{1}{8}\log 8 + \frac{1}{8}\log 8$$

$$= \frac{1}{2} + \frac{1}{2} + \frac{3}{8} + \frac{3}{8} = 1 + \frac{3}{4} = \frac{7}{4} = 1.75$$

∴ H(X) : Binary entropy of the source = 1.75.

ii. Design a Huffman code for the source. (5 points)



∴ The code for the source

a_1	1
a_2	01
a_3	001
a_4	000

Problem 9 (10 points)
Given the Hash Table shown below

Key			
0	0	10	NIL
1	1	21	NIL
2	52	NIL	NIL
3	33	13	23
4	NIL	NIL	NIL

Given the hash function is computed as Index := Input MOD 5

Part a. Show the hash table after inserting element 9 (2 points)

0	0	10	NIL
1	1	21	NIL
2	52	NIL	NIL
3	33	13	23
4	9	NIL	NIL

Part b. Show the hast table after inserting element 53 (use linear probing) (5 points)

0	0	10	NIL
1	1	21	NIL
2	52	NIL	NIL
3	33	13	23
4	9	53	NIL

Part c. Show the hash table after deleting the element 13 (3 points)

0	0	10	NIL
1	1	21	NIL
2	52	NIL	NIL
3	33	NIL	23
4	9	53	NIL

✓ R

Problem 10

State if the following questions are True or False (10 points)

F

F 1. Ada has an built-in garbage collection mechanism (F)

T 2. Hash tables are used when a large number of insertion and deletion operations need to be performed (F)

T 3. A hash table is said to have buckets when an array is used for implementing the table (T)

T 4. Stacks are LIFO structures (T)

F 5. The lowest time complexity for sorting an array using comparisons is O(n log n) (T)

T 6. Linked Lists are static structures (F)

T 7. Matrices are typically two dimensional arrays (T)

T 8. Prefix expressions have the precedence embedded in the expression (T)

F 9. The entropy is bounded by $0 \leq H(x) \leq Log_2(M)$, where M is the maximum number of values X can take, (T)

10. Kruskal's algorithm halts when there are n edges in the MST (where n is number of vertices) (F)