You may be asked to demonstrate/explain your work to the tutor, if you are absent/unavailable or fail to demonstrate properly, zero marks will be awarded.

Text book: Deitel, H M & Deitel, P J 2013, C: How to program, 7th edn, Pearson Prentice-Hall, Upper Saddle River, New Jersey.

1. Please answer the below using examples and diagrams
   a. Explain the differences between an array and a linked list
   b. Describe the steps to insert data at the starting of a linked list.
   c. Describe the steps to insert data at the ending of a linked list.
   d. Describe the steps to delete a node from the linked list.

2. Write a statement or set of statements to accomplish each of the following. Assume that all the manipulations occur in main (therefore, no addresses of pointer variables are needed), and assume the following definitions:

```
struct bankEmployee {
      char name[20];
      int salary;
      struct bankEmployee *next;
};

typedef struct bankEmployee BANKEmployee;
typedef BANKEmployee *BANKEmployeePtr;
```

   a. Create a pointer to the start of the list called `startPtr`. The list is empty.

   b. Create a new node of type `BANKEmployee` that's pointed to by pointer `newPtr` of type `BANKEmployeePtr`. Assign the string `"Justin"` to member `name` and the value `1000` to member salary (use `strcpy`). Provide any necessary declarations and statements.

   c. Assume that the list pointed to by `startPtr` currently consists of 2 nodes one containing `"Justin "` and one containing `"Sam"`. The nodes are in alphabetical order. Provide the statements necessary to insert in order nodes containing the following data for `name` and `salary`:

```
"Antony"        200
"Tony"          300
"Peter"         400
```

   Use pointers `previousPtr`; `currentPtr` and `newPtr` to perform the insertions; State what `previousPtr` and `currentPtr` point to before each insertion. Assume that `newPtr` always points to the new node, and that the new node has already been assigned the data.

   d. Write a while loop that prints the data in each node of the list. Use pointer `currentPtr` to move along the list.

e. Write a `while` loop that deletes all the nodes in the list and frees the memory associated with each node. Use pointer `currentPtr` and pointer `tempPtr` to walk along the list and free memory, respectively.

f. Incorporate all these into a complete C program, your linked list should contain all 5 nodes (Antony, Justin, Peter, Sam, Tony in alphabetical order).

3. Please represent the above Linked list graphically. Need to introduce a new diagram step by step such as creating a new block of memory, assigning null, introducing a current pointer, introducing a previous pointer etc. (This question is very important – Question 2 will not be marked without this question)