**Question 1:**

The following is a skeleton of a selection sort implementation in C.

int minIndex(float d[], int size){ // return the index of the min in the given array

}

void swap(float \*p1, float\* p2) { // swap two vars ...

}

void selectionSort(float d[], int size){

... }

(a) Write C code to implement the above selection sort algorithm.

(b) Evaluate step by step, giving reasons, the time complexity of each of the above functions in terms of the Big-O notation.

**Question 2:**

Following code intends to implement a dynamic stack.

struct node{ float data; struct node\* next; }; struct stack{ struct node\* sp; };

struct node\* makenode(float item){ // make a new node with item ... }

void init(struct stack \* s){...} // initialize sp

int full(struct stack \* s){...} // return 1 if full

int empty(struct stack \* s){...} // return 1 if empty

int push(struct stack \*s, float item){ ... }

float pop(struct stack \*s){ ... }

float top(struct stack \*s){...}

(a) Write a clear diagram to show the status of the stack structure instance, nodes, stored values and node linking after pushing the values 2.0, 6.2 and 7.0.

(b) Write code for each function above to complete the stack implementation.

(c) Write a code to implement a queue data structure. (Follow the same pattern as above)