



**Daffodil**  
*International*  
**University**

## *Final Lab Report -Fall 2022*

***Course Title: Data Structure Lab***

***Course Code: SE-132***

Submitted To:

**Ms. Sazia Sharmin**

**Lecturer (SWE)**

**Daffodil International University**

Submitted By :

**MD : Nurnabi Fahim**

**ID : 221-35-1049**

**Section : E**

**Department : SWE**

**Daffodil International University**

**Github Link:** <https://github.com/fahim1049/Data-Structure-Lab>

### **Code 1.1:**

```
<global> main() : int
problem 1.c
1 //code for insert two element at last of the array
2 #include <stdio.h>
3 #include <stdlib.h>
4 int main() {
5     int *arr, *arr2;
6     int i, n, m;
7     printf("Enter the size of primary size of your array: ");
8     scanf("%d", &n);
9     arr = (int *)malloc(sizeof(int)*n);
10    m = n+2;
11    arr2 = (int *)malloc(sizeof(int)*m);
12    printf("Enter element of your array:\n");
13    for(i = 0; i < n; i++) {
14        scanf("%d", arr+i);
15    }
16    printf("Before insert your elements of your array are:\n\t");
17    for(i = 0; i < n; i++) {
18        printf("%d ", *(arr+i));
19    }
20    for(i = 0 ; i < n ; i++){
21        *(arr2 + i) = *(arr + i);
22    }
23    for(i = n; i < m; i++) {
24        printf("\nEnter a element for insert last: ");
25        scanf("%d", arr2+i);
26    }
27    printf("\nAfter insert two element at last you array is:\n\t");
28    for(i = 0; i < m; i++) {
29        printf("%d ", *(arr2 + i));
30    }
31    return 0 ;
32 }
33
```

## OUTPUT:

```
"C:\Users\Fahim\Desktop\DS report\problem 1.exe"
Enter the size of primary size of your array: 5
Enter element of your array:
2
1
3
4
5
Before insert your elements of your array are:
  2 1 3 4 5
Enter a element for insert last: 6
Enter a element for insert last: 7
After insert two element at last you array is:
  2 1 3 4 5 6 7
Process returned 0 (0x0)   execution time : 47.068 s
Press any key to continue.
```

**Discussion:** Here, at first this programme will take a number as a initial size of an array. Then it integer value as the element of the array. After that, again it takes 2 more value one by one for add at last of the array. At last it will print the final value.

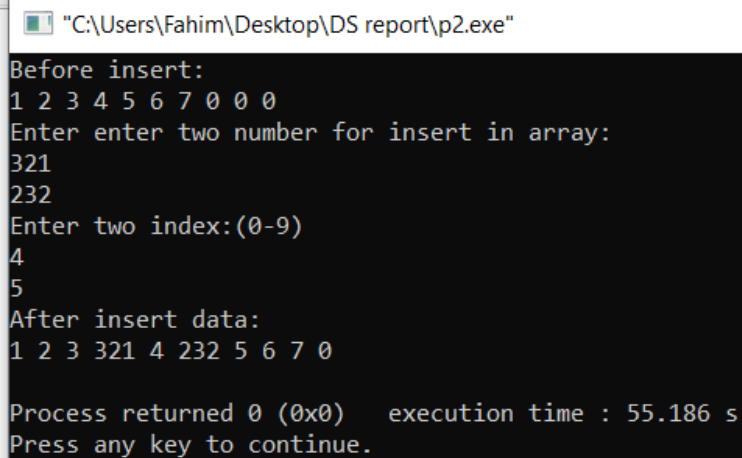
## Code 2.2:

```
Start here x problem 1.c x p2.c x
1
2 //insert two differnt numbers at differernt porsition
3 #include <stdio.h>
4 int arr[10] = {1, 2, 3, 4, 5, 6, 7};
5 void insert_element(int pos, int data);
6 void print();
7 int main() {
8     int pos1, pos2, item1, item2;
9     printf("Before insert:\n");
10    print();
11    printf("Enter enter two number for insert in array:\n");
12    scanf("%d %d", &item1, &item2);
13    printf("Enter two index: (0-9)\n");
14    scanf("%d %d", &pos1, &pos2);
15    if(pos1 > pos2) {
16        insert_element(pos1, item1);
17        insert_element(pos2, item2);
18    }
19    else {
20        insert_element(pos2, item2);
21        insert_element(pos1, item1);
22    }
23    printf("After insert data:\n");
24    print();
25    return 0;
26 }
27 void insert_element(int pos, int data) {
28     int i;
29     for(i = 9; i >= pos; i--) {
30         arr[i] = arr[i-1];
31     }
32     arr[i] = data;
33 }
```

```
34 void print() {  
35     int i;  
36     for(i = 0; i < 10; i++) {  
37         printf("%d ", arr[i]);  
38     }  
39     printf("\n");  
40 }  
41
```

---

## **OUTPUT :**



```
"C:\Users\Fahim\Desktop\DS report\p2.exe"  
Before insert:  
1 2 3 4 5 6 7 0 0 0  
Enter enter two number for insert in array:  
321  
232  
Enter two index:(0-9)  
4  
5  
After insert data:  
1 2 3 321 4 232 5 6 7 0  
  
Process returned 0 (0x0)   execution time : 55.186 s  
Press any key to continue.
```

---

## ***Discussion:***

At first, this program will show you some element of an array which was predefined. Secondly it will asked for two more value and 2 position between zero(0) to nine and last it just add those value in those position and print the array again

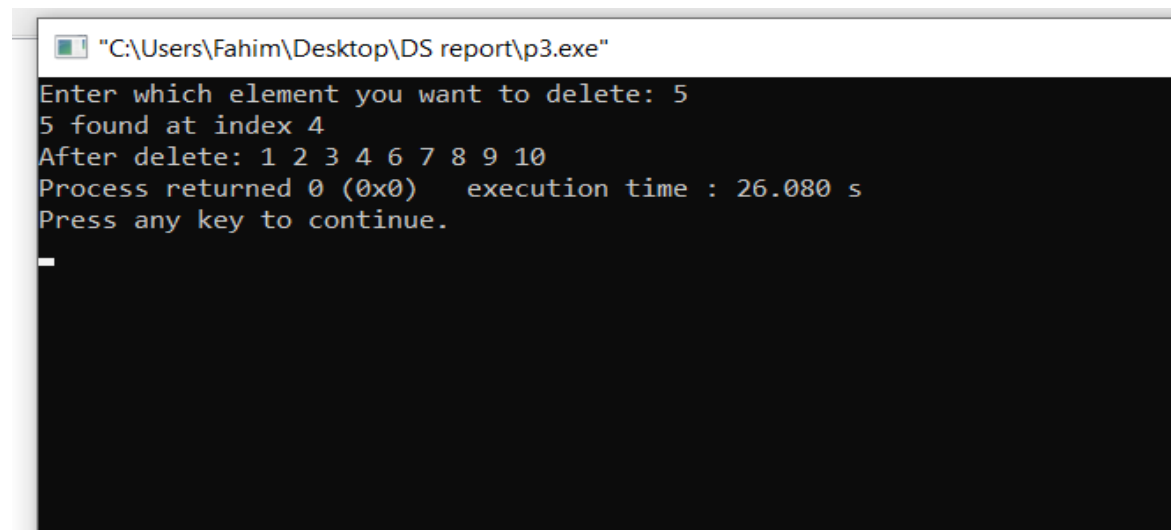
### **Code 3.3:**

```
Start here x problem 1.c x p2.c x p3.c x
1 //applying binary search & delete
2 #include <stdio.h>
3 #include <stdbool.h>
4 #include <stdlib.h>
5 int main() {
6     int min = 0, max = 9, mid, del_item, i;
7     bool flag = true;
8     int arr[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
9     printf("Enter which element you want to delete: ");
10    scanf("%d", &del_item);
11    while(min <= max) {
12        mid = (min + max) / 2;
13        if(del_item == arr[mid]) {
14            printf("%d found at index %d\n", del_item, mid);
15            flag = false;
16            break;
17        }
18        else if(arr[mid] < del_item) {
19            min = mid + 1;
20        }
21        else {
22            max = mid - 1;
23        }
24    }
25    if(flag) {
26        printf("%d is not available in the data set\n", del_item);
27    }
28    else {
29        for(i = mid; i < 9; i++) {
30            arr[i] = arr[i+1];
31        }

```

```
31 }
32 printf("After delete: ");
33 for(i = 0; i < 9; i++) {
34     printf("%d ", arr[i]);
35 }
36 }
37 return 0;
38 }
39
```

### **Output:**



```
"C:\Users\Fahim\Desktop\DS report\p3.exe"
Enter which element you want to delete: 5
5 found at index 4
After delete: 1 2 3 4 6 7 8 9 10
Process returned 0 (0x0)   execution time : 26.080 s
Press any key to continue.
_
```

### **Discussion:**

At first, this programme will ask you for a integer value to apply binary search. If it is exist in the array user will see a message " \_\_\_\_ found at index \_\_\_\_" and the data will deleted and print all the data which are exist after deletion. Otherwise it will print another message " \_\_\_\_ is not available in the data set"

## Code 4.1 :

```
Start here x problem 1.c x p2.c x p3.c x p4.c x
1 //comparing linear and binary serch
2 #include <stdio.h>
3 #include <stdbool.h>
4 int main() {
5     int arr[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
6     int max, min, mid, count_binary = 0, count_linear = 0;
7     int search_item;
8     bool flag = true;
9     printf("Enter which item you want to search(1-10): ");
10    scanf("%d", &search_item);
11    printf("\n");
12    //for bianary
13    min = 0;
14    max = 9;
15    while(min <= max) {
16        count_binary++;
17        mid = (min + max) / 2;
18        if(arr[mid] == search_item) {
19            printf("%d found after looping %d times\n", search_item,
20                count_binary);
21            flag = false;
22            break;
23        }
24        else if(search_item < arr[mid]) {
25            max = mid - 1;
26        }
27        else {
28            min = mid + 1;
29        }
30    }
```

```

32 for(int i = 0; i < 10; i++) {
33     count_linear++;
34     if(arr[i] == search_item) {
35         printf("%d found after looping %d time(s)\n", search_item,
36             count_linear);
37         break;
38     }
39 }
40 if(flag) {
41     printf("Data is not available\n");
42 }
43 else {
44     printf("\n\n====>\t\n\n");
45 }
46 return 0;
47 }
48

```

---

### **Output:**

```

"C:\Users\Fahim\Desktop\DS report\p4.exe"
Enter which item you want to search(1-10): 9

9 found after looping 3 times
9 found after looping 9 time(s)

.
t====>

Process returned 0 (0x0)   execution time : 20.587 s
Press any key to continue.

```

---

### **Discussion :**



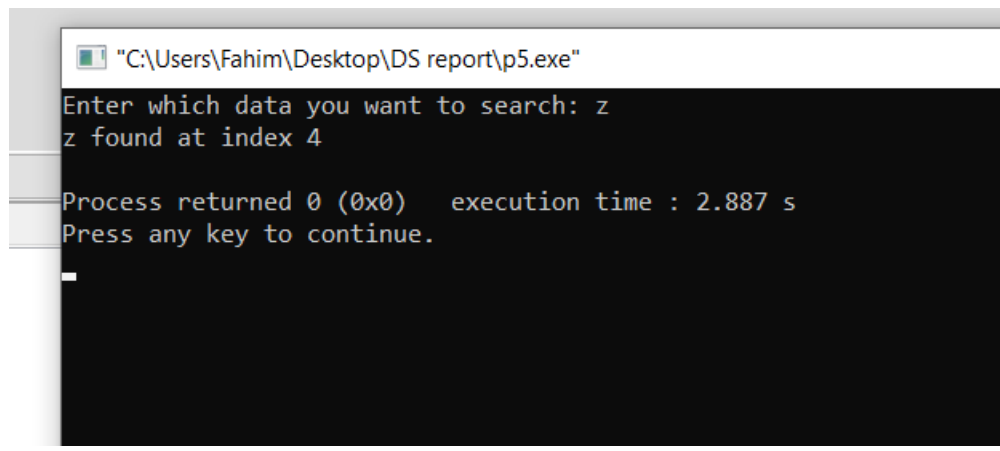
At first, this program want to know about one number 0-10. Then it will start a both search respectively and compare both by counting loop. And so a message based on the count result.

### **Code 5.1 :**

```
Start here x problem 1.c x p2.c x p3.c x p4.c x p5.c x
1 //Short and binary search
2 #include <stdio.h>
3 #include <stdbool.h>
4 int main() {
5     char arr[5] = {'z', 'k', 'l', 'a', 'g'};
6     int i, j, min, max, mid;
7     char temp, search_data;
8     bool flag = true;
9     for(i = 0; i < 5; i++) {
10         for(j = 0; j < (4 - i); j++) {
11             if(arr[j] > arr[j+1]) {
12                 temp = arr[j];
13                 arr[j] = arr[j+1];
14                 arr[j+1] = temp;
15             }
16         }
17     }
18     printf("Enter which data you want to search: ");
19     fflush(stdin);
20     scanf("%c", &search_data);
21     min = 0;
22     max = 4;
23     while (min <= max) {
24         mid = (min + max) / 2;
25         if(arr[mid] == search_data) {
26             printf("%c found at index %d\n", search_data, mid);
27             flag = false;
28             break;
29         }
30         else if(search_data < arr[mid]) {
31             max = mid - 1;
32         }
33         else {
```

```
34     min = mid + 1;
35     }
36     }
37     if(flag) {
38         printf("Data is not in the array\n");
39     }
40     return 0;
41     }
42 }
```

**Output:**



```
"C:\Users\Fahim\Desktop\DS report\p5.exe"
Enter which data you want to search: z
z found at index 4

Process returned 0 (0x0)   execution time : 2.887 s
Press any key to continue.
_
```

Firstly, this programme will sort it's given data and ask for a character to Search in the array. If match it'll show a message "--- found at----".

Otherwise, it'll show "Data is not in the array"

## **Discussion :**

Firstly, this programme will sort it's given data and ask for a character to Search in the array. If match it'll show a message "--- found at----". Otherwise, it'll show "Data is not in the array"

## **Code 6.1:**

Start here X problem 1.c X p2.c X p3.c X p4.c X p5.c X p6.c X

```
1 //stack
2 #include <stdio.h>
3 #include <conio.h>
4 #include <stdlib.h>
5 #include <stdbool.h>
6 char arr[5];
7 int top = -1;
8 void option();
9 void insert_stack(char data);
10 void delete();
11 void print();
12 int main() {
13     int data, choice;
14     while(true) {
15         option();
16         scanf("%d", &choice);
17         switch(choice) {
18             case 1:
19                 printf("Enter a character for add in stack: ");
20                 fflush(stdin);
21                 scanf("%c", &data);
22                 insert_stack(data);
23                 break;
24             case 2:
25                 delete();
26                 break;
27             case 3:
28                 print();
29                 break;
30             case 4:
31                 printf("STOPED\n");
32                 exit(0);
33             default:
```

```
33     default:
34         printf("Invalid input.....\n");
35         fflush(stdin);
36         getchar();
37     }
38 }
39 return 0;
40 }
41 void option() {
42     system("cls");
43     printf("1. insert\n");
44     printf("2. delete\n");
45     printf("3. print\n");
46     printf("4. exit\n");
47     printf("\nChoice option: ");
48 }
49 void insert_stack(char data) {
50     top++;
51     if (top >= 5) {
52         printf("====>Data overflow!!!!!!!!\n");
53         fflush(stdin);
54         getchar();
55         top = 4;
56     }
57     else {
58         arr[top] = data;
59     }
60 }
61 void delete() {
62     if (top <= -1) {
63         printf("====>Data underflow!!!!!!!!\n");
```

---

```

59     }
60 }
61 void delete() {
62     if(top <= -1) {
63         printf("====>Data underflow!!!!!!\n");
64         fflush(stdin);
65         getchar();
66     }
67     else {
68         printf("%c have deleted\n", arr[top]);
69         fflush(stdin);
70         getchar();
71         arr[top] = NULL;
72         top--;
73     }
74 }
75 void print() {
76     int i;
77     if(top == -1) {
78         printf("Empty stack\n");
79     }
80     else {
81         printf("Data: ");
82         for(i = top; i >= 0; i--) {
83             printf("%c", arr[i]);
84         }
85     }
86     fflush(stdin);
87     getchar();
88 }
89

```

## OUTPUT:

```

1. insert
2. delete
3. print
4. exit

```

Choice option: 1

Enter a character for add in stack: E

```

1. insert
2. delete
3. print
4. exit

```

Choice option: 1

Enter a character for add in stack: L

1. insert
2. delete
3. print
4. exit

Choice option: 1

Enter a character for add in stack: P

1. insert
2. delete
3. print
4. exit

Choice option: 1

Enter a character for add in stack: P

1. insert
2. delete
3. print
4. exit

Choice option: 1

Enter a character for add in stack: P

**Continue.....**

**Discussion:**

This programme is about stack which flow the LIFO method, means Last In First Out. At first, this programme will show us 4 option and ask for choice one of 1. Is for add data in stack 2. is for delete from stack 3. is for print data from stack 4. is for terminate the programme This stack will contain max five element at a time if anybody want to add more than five elements then the programme will show him/her this message “=====>Data overflow!!!!!!!!!!” and also if there will no data in the stack and anyone want to delete from it then he/she will see this message “=====>Data underflow!!!!!!!!!!”.

## Code 7.1:

```
<global> main(): int
Start here X problem 1.c X p2.c X p3.c X p4.c X p5.c X p6.c X p7.c X
1 //Creating Linked list:
2 #include <stdio.h>
3 #include <stdlib.h>
4 typedef struct node NODE;
5 struct node {
6     char ch;
7     NODE *next;
8 };
9 NODE *create_node(char data);
10 NODE *insert_last(NODE *head, char data);
11 void print(NODE *head);
12 int main() {
13     char item;
14     int choice = 1;
15     NODE *head = NULL;
16     while (choice) {
17         printf("Enter a character to add in linked list: ");
18         fflush(stdin);
19         scanf("%c", &item);
20         head = insert_last(head, item);
21         printf("Do you want to continue \n(press zero(0) for exit:");
22         scanf("%d", &choice);
23     }
24     printf("Your linked list is: ");
25     print(head);
26     printf("\nEnter new character to insert at last of your list: ");
27     fflush(stdin);
28     scanf("%c", &item);
29     head = insert_last(head, item);
30     printf("\nAfter insertion your list is: ");
31     print(head);
32     return 0;
33 }
```

```
33 }
34 NODE *create_node(char data) {
35     NODE *new_node = (NODE*)malloc(sizeof(NODE));
36     if (new_node != NULL) {
37         new_node->ch = data;
38         new_node->next = NULL;
39     }
40     return new_node;
41 }
42 NODE *insert_last(NODE *head, char item) {
43     NODE *current_node = create_node(item);
44     if (head == NULL) {
45         return current_node;
46     }
47     NODE *temp = head;
48     while (temp->next != NULL) {
49         temp = temp->next;
50     }
51     temp->next = current_node;
52     return head;
53 }
54 void print(NODE *head) {
55     if (head == NULL) {
56         printf("List is empty!\n");
57     }
58     else {
59         NODE *temp = head;
60         while (temp != NULL) {
61             printf("[%x]==>[%c]", temp, temp->ch);
62             temp = temp->next;
63         }
64     }
65 }
```



```

51     temp->next = current_node;
52     return head;
53 }
54 void print(NODE *head) {
55     if(head == NULL) {
56         printf("List is empty!\n");
57     }
58     else {
59         NODE *temp = head;
60         while(temp != NULL) {
61             printf("[%x]====>[%c]", temp, temp->ch);
62             temp = temp->next;
63         }
64         printf("[NULL]\n");
65     }
66 }
67

```

## OUTPUT:

```

"C:\Users\Fahim\Desktop\DS report\p7.exe"
Enter a character to add in linked list: A
Do you want to continue
(press zero(0) for exit:1
Enter a character to add in linked list: B
Do you want to continue
(press zero(0) for exit:1
Enter a character to add in linked list: C
Do you want to continue
(press zero(0) for exit:0
Your linked list is: [b26a10]====>[A][b26a30]====>[B][b26a50]====>[C][NULL]
;
Enter new character to insert at last of your list: D
After isertion your list is: [b26a10]====>[A][b26a30]====>[B][b26a50]====>[C][b26a70]====>[D][NULL]
Process returned 0 (0x0)   execution time : 19.632 s
Press any key to continue.

```

## Discussion:

At first, this programme will ask for enter a charackter to insert in linked list. And ask user is he/she want to continue? If he/she want to continue again it will ask for a character and it will continue while he/she press zero (0). If press 0 then it will print all the value he/she input and it's address in hexadecimal. After that it will be wait for again give a data for insert a last and again it print life previous with new data. Finally, it will terminate.

## Code8.1:

```
Start here X problem 1.c X p2.c X p3.c X p4.c X p5.c X p6.c X p7.c X p8.c X
1 //implementation of queue
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include <stdbool.h>
5 typedef struct node Q_NODE;
6 struct node {
7     int data;
8     Q_NODE *next;
9 };
10 typedef struct counter_node C_NODE;
11 struct counter_node {
12     int tail;
13 };
14 Q_NODE *create_node(int data);
15 Q_NODE *add_node_last(Q_NODE *head, int data);
16 Q_NODE *delete_first(Q_NODE *head);
17 void print(Q_NODE *head);
18 void option();
19 int main() {
20     Q_NODE *head = NULL;
21     C_NODE count;
22     count.tail = 0;
23     int q_size, optn, data;
24     printf("Enter the size of your Queue: ");
25     scanf("%d", &q_size);
26     while(true){
27         option();
28         printf("\n\nchoice your option: ");
29         scanf("%d", &optn);
30         switch(optn) {
31             case 1:
32                 {
33                     printf("Enter a data for insert: ");
34                     scanf("%d", &data);
35                     count.tail++;
36                     if(count.tail > q_size) {
37                         printf("Queue is full. Data overFlow\n");
38                         count.tail--;
39                         printf("\n\npress any key to Countinue\n\n");
40                         fflush(stdin);
41                         getchar();
42                     }
43                     else {
44                         head = add_node_last(head, data);
45                     }
46                 }
47                 break;
48             case 2:
49                 {
50                     count.tail--;
51                     if(count.tail < 0) {
52                         count.tail++;
53                     }
54                 }
55                 break;
56             case 3:
57                 {
58                     print(head);
59                 }
60                 break;
61             case 4:
62                 {
63                     delete_first(head);
64                 }
65                 break;
66             case 5:
67                 {
68                     option();
69                 }
70                 break;
71             default:
72                 {
73                     printf("Invalid option\n");
74                 }
75                 break;
76         }
77     }
78 }
```

```

53  if(count.tail < 0) {
54      count.tail++;
55  }
56  else {
57      head = delete_first(head);
58  }
59  }
60  printf("\n\npress any key to continue.....");
61  fflush(stdin);
62  getchar();
63  break;
64  case 3:
65  printf("You have: ");
66  print(head);
67  break;
68  case 4:
69  exit(0);
70  default:
71  printf("Invalid input.....\n\n\n");
72  }
73  }
74  return 0;
75  }
76  void option() {
77  printf("1. Insert\n");
78  printf("2. Delete\n");
79  printf("3. Print queue\n");
80  printf("4. Exit");
81  }
82  Q_NODE *create_node(int data) {

```

---

```

81     }
82     Q_NODE *create_node(int data) {
83         Q_NODE *new_node = (Q_NODE*)malloc(sizeof(Q_NODE));
84         if(new_node != NULL) {
85             new_node->data = data;
86             new_node->next = NULL;
87         }
88         return new_node;
89     }
90     Q_NODE *add_node_last(Q_NODE *head, int data) {
91         Q_NODE *current_node = create_node(data);
92         if(head == NULL) {
93             return current_node;
94         }
95         Q_NODE *temp = head;
96         while(temp->next != NULL) {
97             temp = temp->next;
98         }
99         temp->next = current_node;
100         return head;
101     }
102     Q_NODE *delete_first(Q_NODE *head) {
103         if(head == NULL) {
104             printf("\nData underflow\n");
105         }
106         else {
107             printf("%d is deleted\n", head->data);
108             head = head->next;
109         }
110         return head;
111     }
112     void print(Q_NODE *head) {

```

---

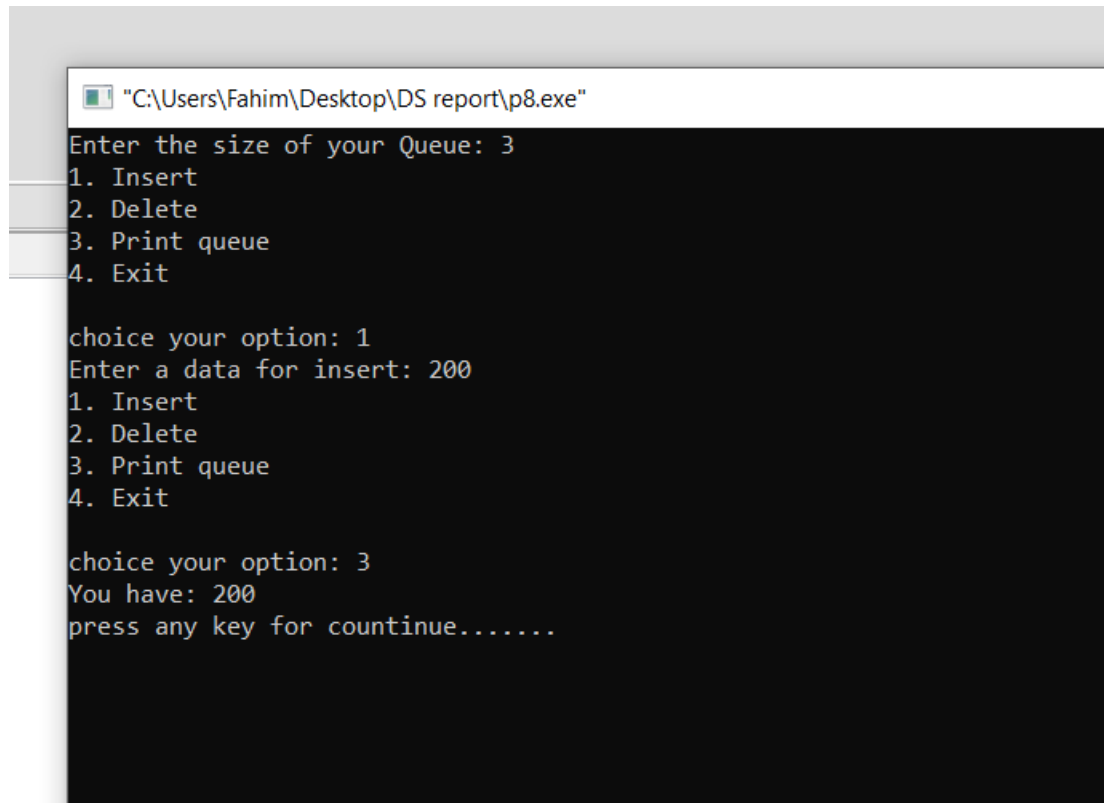
```

113         if(head == NULL) {
114             printf("Empty Queue\n");
115         }
116         else {
117             Q_NODE *temp = head;
118             while(temp != NULL) {
119                 printf("%d ", temp->data);
120                 temp = temp->next;
121             }
122         }
123         printf("\n");
124         printf("press any key for countinue.....\n\n");
125         fflush(stdin);
126         getchar();
127     }

```

---

**Output:**

A screenshot of a Windows command prompt window. The title bar shows the file path "C:\Users\Fahim\Desktop\DS report\p8.exe". The command prompt displays the following text: "Enter the size of your Queue: 3", followed by a menu: "1. Insert", "2. Delete", "3. Print queue", "4. Exit". The user has entered "1", and the prompt shows "choice your option: 1". Then, the user enters "200", and the prompt shows "Enter a data for insert: 200". The menu is shown again, and the user enters "3", with the prompt showing "choice your option: 3". Finally, the prompt shows "You have: 200" and "press any key for countinue.....".

```
"C:\Users\Fahim\Desktop\DS report\p8.exe"
Enter the size of your Queue: 3
1. Insert
2. Delete
3. Print queue
4. Exit

choice your option: 1
Enter a data for insert: 200
1. Insert
2. Delete
3. Print queue
4. Exit

choice your option: 3
You have: 200
press any key for countinue.....
```

### **Discussion:**

This programme is follow queue structure which is follow FIFO, means First In First Out method. At first, this programme will show a message for take the size of the Queue. Then it will show us 4 option and ask for choice one of 1. Is for add data in stack 2. is for delete from stack 3. is for print data from stack 4. is for terminate the programme Similarly, like stack it is option are work and this program also handle the overflow and underflow.

## Code 9.1:

```
Start here X problem 1.c X *p2.c X p3.c X p4.c X p5.c X p6.c X p7.c X p8.c X p9.c X
1
2 //binary search tree
3 #include <stdio.h>
4 #include <stdlib.h>
5 #include <stdbool.h>
6 typedef struct node tree_node;
7 struct node {
8     char value;
9     tree_node *left;
10    tree_node *right;
11 };
12 //all function declares here
13 tree_node *create_node(char item);
14 tree_node *add_node(tree_node *root, char item);
15 void print_pre(tree_node *root);
16 void print_in(tree_node *root);
17 void print_post(tree_node *root);
18 //main function
19 int main() {
20     tree_node *root = NULL;
21     int choice;
22     char item;
23     while(true) {
24         printf("Enter a data to add in tree: ");
25         fflush(stdin);
26         scanf("%c", &item);
27         root = add_node(root, item);
28         printf("Do you want to continue: ");
29         scanf("%d", &choice);
30         if(!choice) {
31             break;
```

```

32     }
33 }
34 printf("Pre order: ");
35 print_pre(root);
36 printf("\nInorder: ");
37 print_in(root);
38 printf("\nPost Order: ");
39 print_post(root);
40 return 0;
41 }
42 //node create function
43 tree_node *create_node(char item) {
44     tree_node *new_node = (tree_node*)malloc(sizeof(tree_node));
45     if(new_node != NULL) {
46         new_node->value = item;
47         new_node->left = NULL;
48         new_node->right = NULL;
49     }
50     return new_node;
51 }
52 //node add function
53 tree_node *add_node(tree_node *root, char item){
54     tree_node *temp = root;
55     tree_node *current_node = create_node(item);
56     if(temp == NULL) {
57         return current_node;
58     }
59     if(item < temp->value) {
60         temp->left = add_node(temp->left, item);
61     }
62     else {
63         temp->right = add_node(temp->right, item);

```

---

```

64     }
65     return temp;
66 }
67 //print pre order function
68 void print_pre(tree_node *root) {
69     tree_node *temp = root;
70     if(temp == NULL) {
71         return;
72     }
73     printf("%c ", temp->value);
74     print_pre(temp->left);
75     print_pre(temp->right);
76 }
77 //print in order function
78 void print_in(tree_node *root) {
79     tree_node *temp = root;
80     if(temp == NULL) {
81         return;
82     }
83     print_in(temp->left);
84     printf("%c ", temp->value);
85     print_in(temp->right);
86 }
87 //print post order function
88 void print_post(tree_node *root) {
89     tree_node *temp = root;
90     if(temp == NULL) {
91         return;
92     }
93     print_post(temp->left);

```

```

94     print_post(temp->right);
95     printf("%c ", temp->value);
96 }
97 //print post order function
98 void print_post(tree_node *root) {
99     tree_node *temp = root;
100    if(temp == NULL) {
101        return;
102    }
103    print_post(temp->left);
104    print_post(temp->right);
105    printf("%c ", temp->value);
106 }

```



## **Output:**

```
Enter a data to add in tree: 50
Do you want to continue: 1
Enter a data to add in tree: 40
Do you want to continue: 1
Enter a data to add in tree: 70
Do you want to continue: 1
Enter a data to add in tree: 35
Do you want to continue: 1
Enter a data to add in tree: 45
Do you want to continue: 1
Enter a data to add in tree: 60
Do you want to continue: 1
Enter a data to add in tree: 75
Do you want to continue: 0
Pre order: 50 40 35 45 70 60 75
Inorder: 35 40 45 50 60 70 75
Post Order: 35 45 40 60 75 70 50
```

## **Discussion:**

This programme is as similar as 9.1. The difference between this and previous programme is it is work with integer and previously one is with character. That's is