1.main.c

```
#include "header.h"


int main(){
    stack s;
    int choice, element, size, index;
    while(1){
        printf("\nStack Menu\n");
        printf("1. init\n");
        printf("2. Push\n");
        printf("3. Pop\n");
        printf("4. Stack Top\n");
        printf("5. Peek at specific index\n");
        printf("6. Display\n");
        printf("7. Exit\n");
        printf("Enter Your Choice: ");
        scanf("%d", &choice);
        switch(choice){
            case 1:
            printf("Enter the size of the stack: ");
                scanf("%d", &size);
                init(&s, size);
                break;
            case 2:
                if(!isFull(s)){
                    printf("Enter the element to push: ");
                    scanf("%d", &element);
                    push(&s, element);
                }
                else{
                    printf("Stack is Full\n");
    }
    break;
case 3:
    if(!isEmpty(s)){
        printf("Popped element is: %d\n", pop(&s));
    }
    else{
        printf("Stack is empty\n");
    }
    break;
case 4:
    if(!isEmpty(s)){
        printf("Stack Top element is: %d\n", stackTop(s));
    }
    else{
        printf("Stack is empty\n");
    }
    break;
case 5:
    if(!isEmpty(s)){
        printf("Enter the index to peek from stack: ");
        scanf("%d", &index);
        peek(&s, index);
    }
    else{
        printf("Stack is empty\n");
    }
    break;
case 6:
    display(s);
    break;
```

```
            case 7:
                printf("Exiting...\n");
                free(s.a);
                exit(0);
                break;
            default:
                printf("Invalid Choice, Try again!\n");
        }
    }

    return 0;
}
```

2.header.h

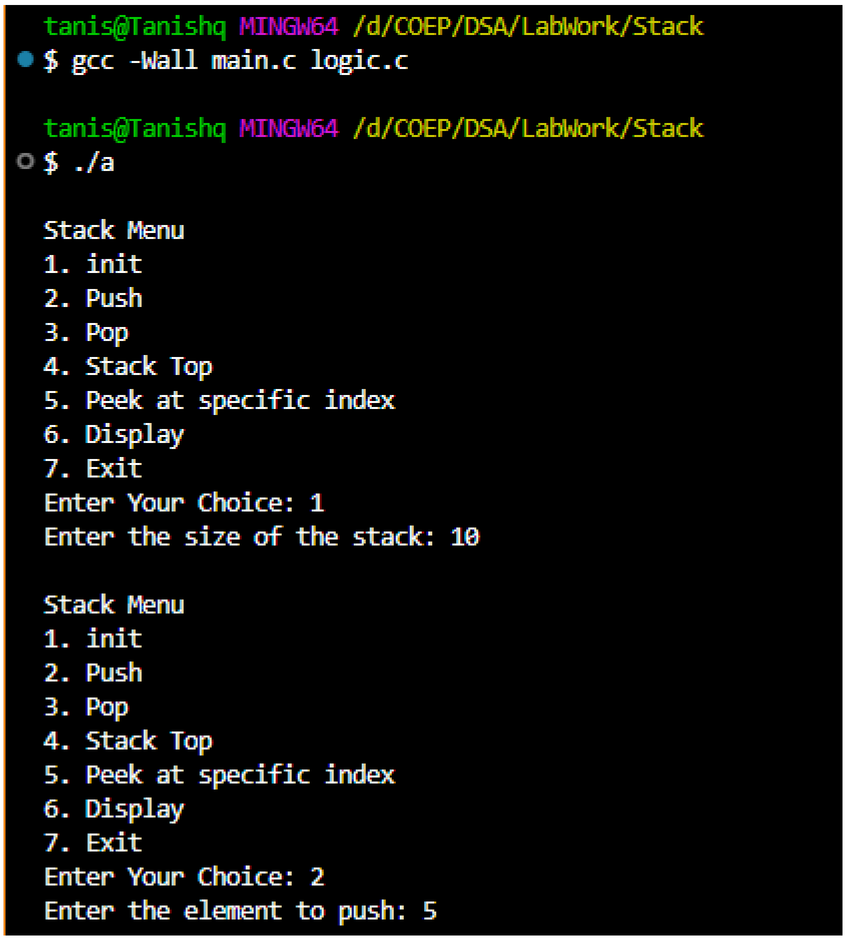
```
#include <stdio.h>

#include <stdlib.h>


typedef struct stack{

   int top;

   int size;

   int *a;

}stack;


void init(stack *s, int size);

int isFull(stack s);

int isEmpty(stack s);

void push(stack *s, int data);

int pop(stack *s);

int stackTop(stack s);

void display(stack s);

void peek(stack *s, int index);
```

3.logic.c

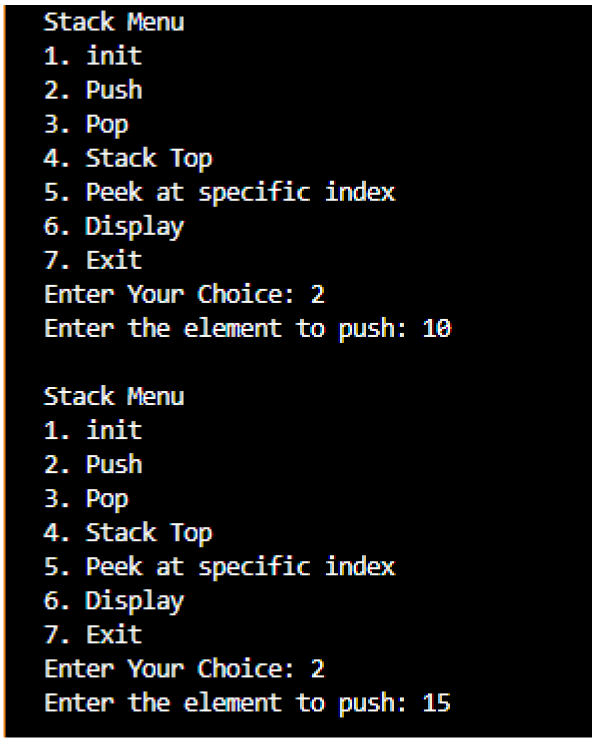
```
#include "header.h"


void init(stack *s, int size){
   s -> size = size;
   s -> top = -1;
   s -> a = (int *)malloc(sizeof(int)*size);
}
int isFull(stack s){
   if(s.top == s.size - 1){
      return 1;
   }
   else{
      return 0;
   }
}
int isEmpty(stack s){
   if(s.top == -1){
      return 1;
   }
   else{
      return 0;
   }
}
void push(stack *s, int data){
   if(isFull(*s)){
      printf("Stack is Full\n");
      return;
   }
   s -> top++;
   s -> a[s -> top] = data;
}
int pop(stack *s){
   if(isEmpty(*s)){
      printf("Stack is Empty\n");
      return -1;
   }
   else{
      int popped = s -> a[s -> top];
      s -> top--;
      return popped;
   }
}
int stackTop(stack s){
   if(isEmpty(s)){
      printf("Stack is Empty\n");
      return -1;
   }
   return s.a[s.top];
}
void display(stack s){
   if(isEmpty(s)){
      printf("Stack is empty\n");
   }
   else{
      int i;
      for(i = s.top; i >= 0; i--){
         printf("%d ", s.a[i]);
      }
      printf("\n");
   }
}
```

```
void peek(stack *s, int index){
   if(index <= 0 || index > s -> top + 1){
      printf("Invalid Index\n");
   }
   else{
      int x = s -> a[s -> top - index + 1];
      printf("Element at index %d from the top: %d\n", index, x);
   }
}
```

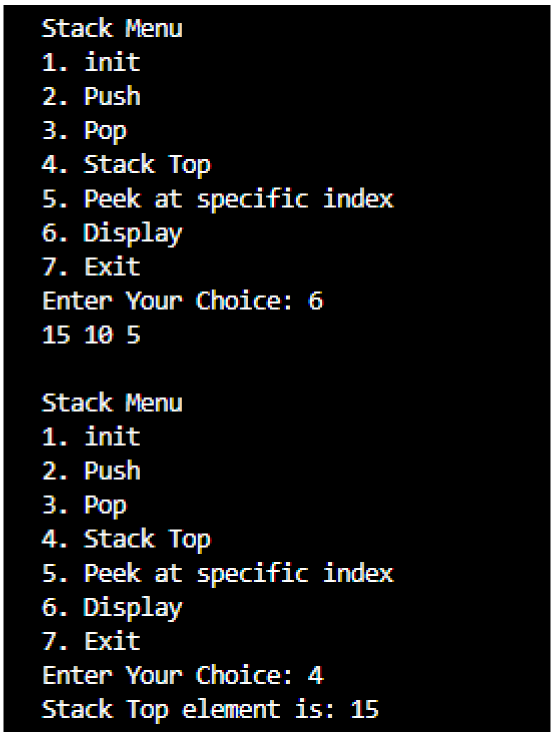
OUTPUT:



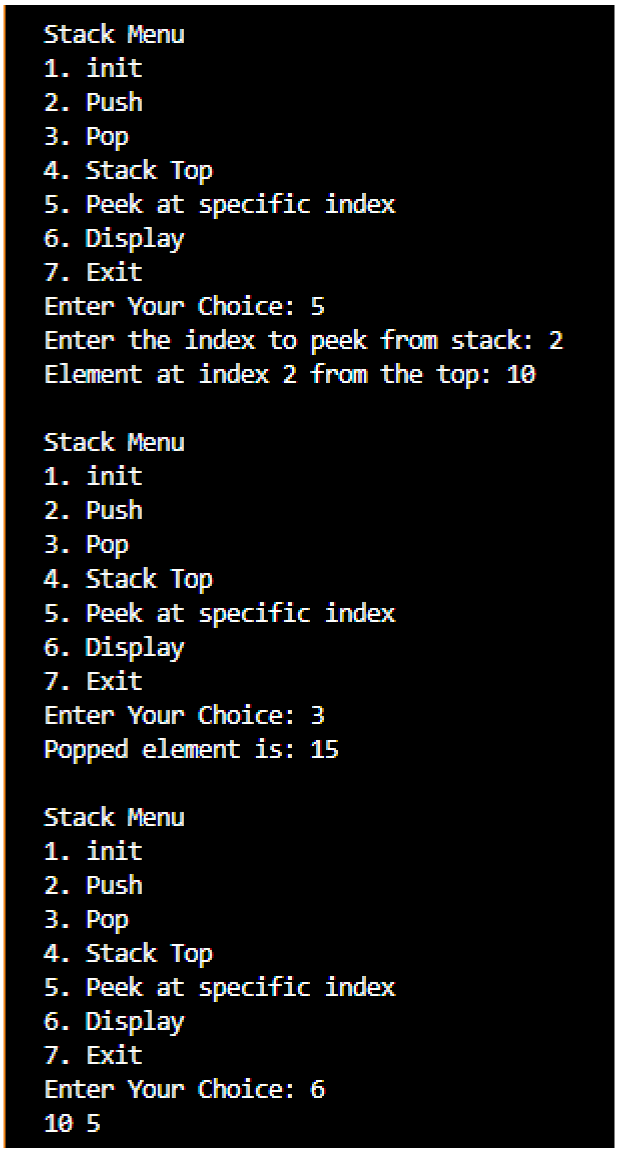
```
tanis@Tanishq MINGW64 /d/COEP/DSA/LabWork/Stack
$ gcc -Wall main.c logic.c

tanis@Tanishq MINGW64 /d/COEP/DSA/LabWork/Stack
$ ./a

Stack Menu
1. init
2. Push
3. Pop
4. Stack Top
5. Peek at specific index
6. Display
7. Exit
Enter Your Choice: 1
Enter the size of the stack: 10

Stack Menu
1. init
2. Push
3. Pop
4. Stack Top
5. Peek at specific index
6. Display
7. Exit
Enter Your Choice: 2
Enter the element to push: 5
```



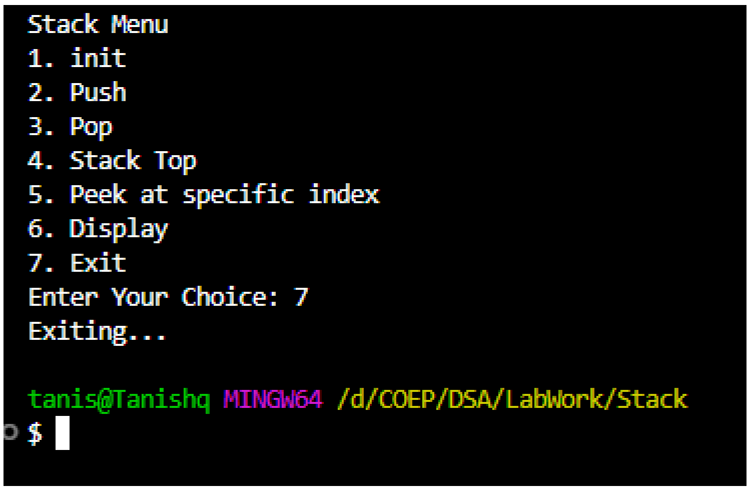
```
Stack Menu
1. init
2. Push
3. Pop
4. Stack Top
5. Peek at specific index
6. Display
7. Exit
Enter Your Choice: 2
Enter the element to push: 10

Stack Menu
1. init
2. Push
3. Pop
4. Stack Top
5. Peek at specific index
6. Display
7. Exit
Enter Your Choice: 2
Enter the element to push: 15
```



```
Stack Menu
1. init
2. Push
3. Pop
4. Stack Top
5. Peek at specific index
6. Display
7. Exit
Enter Your Choice: 6
15 10 5

Stack Menu
1. init
2. Push
3. Pop
4. Stack Top
5. Peek at specific index
6. Display
7. Exit
Enter Your Choice: 4
Stack Top element is: 15
```



```
Stack Menu
1. init
2. Push
3. Pop
4. Stack Top
5. Peek at specific index
6. Display
7. Exit
Enter Your Choice: 5
Enter the index to peek from stack: 2
Element at index 2 from the top: 10

Stack Menu
1. init
2. Push
3. Pop
4. Stack Top
5. Peek at specific index
6. Display
7. Exit
Enter Your Choice: 3
Popped element is: 15

Stack Menu
1. init
2. Push
3. Pop
4. Stack Top
5. Peek at specific index
6. Display
7. Exit
Enter Your Choice: 6
10 5
```



```
Stack Menu
1. init
2. Push
3. Pop
4. Stack Top
5. Peek at specific index
6. Display
7. Exit
Enter Your Choice: 7
Exiting...

tanis@Tanishq MINGW64 /d/COEP/DSA/LabWork/Stack
$
```