

**SUPREME KNOWLEDGE FOUNDATI0N GROUP**

**OF INSTITUTIONS**

NAME : ABHIJIT PAL

UNIVERSITY ROLL NO : 25300121050

UNIVERSITY REG. NO: 212530100110052

STREAM : CSE YEAR : 2nd SEMISTER : 3rd

SUBJECT NAME : Data Structure& Algorithm

Lab

SUBJECT CODE : PCC-CS391

**ASSIGNMENT-3**

**Date-12/08/2022**

1. **Write a program to implement stack ADT.**
   1. Push
   2. Pop
   3. Display
   4. Peek
   5. Isempty
   6. Isfull
   7. Reverse
   8. Count
   9. Searching

**ALGORITHM :**

**STEP 1 : START.**

**STEP 2 :** Define the Array size.

**STEP 3 :** Declare the variable top,stack[max].

**STEP 4 :** Set top=-1.

**STEP 5 :** Declare the function name

Void push[]

Void pop[]

Void display[]

Void isEmpty[]

Void isFull[]

Void reverse[]

Void isCount[]

Void search[]

**STEP 6 :** Start the main function and Declare variable ch .

**STEP 7 :** Then Operation will start using switch case.

**STEP 8 :** For push Function :-

STEP 8.1 : Declare a variable val.

STEP 8.2 : if(top==max-1)

->Print Stack is Full

Else

->print the element to push and hold it in var variable.

-> top=top+1.

->stack[top]=val.

**STEP 9 :** For pop Function :-

STEP 9.1 : if(top==-1)

-> print that Stack is empty .

Else

-> print stack[top] .

-> top=top-1;

**STEP 10 :** For display Function.

STEP 10.1 : Declare a variable I .

STEP 10.2 : if(top==-1) .

-> print that Stack is empty .

Else

-> print Stack Using for loop.

-> print stack[i] .

**STEP 11 :** For isEmpty Function .

STEP 11.1 : if(top == -1)

-> print Stack is empty .

-> return 1;

-> print that Stack is not empty .

-> return 0 .

**STEP 12 :** For isFull Funtcion -:

STEP 12.1 : if(top ==MAX-1)

-> print that Stack is full .

-> return 1 .

-> print that Stack Is Not full .

-> return 0 .

**STEP 13 :** For peek Function.

STEP 13.1 : Declare a variable x .

STEP 13.2 : Set x= stack[top] .

-> print the top most element.

-> return x .

**STEP 14 :** For reverse Function :-

STEP 14.1 : Declare a variable i .

STEP 14.2 : if(top==-1)

-> print that Stack is Empty .

Else

-> Print the stack[i] by Using For loop.

**STEP 15 :** For iscount Function :-

STEP 15.1 : if(top==-1)

-> print that Element in stack -: 0.

Else

-> print Element in the stack - (top+1) .

**STEP 16 :** For search Function .

STEP 16.1 : Declare the variable x,i,f=0 .

-> print Enter the element in the stack to search

and hold it by scanf function.

STEP 16.2 : Set i=0 .

-> while(i<=top)

-> if(stack[i]==x)

-> f=1 .

-> Then Increase the value of I .

-> if(f==1)

-> print item found by using x variable .

Else

-> print item is not found .

**STEP 17 : END .**

**CODE**

#include<stdio.h>

#define MAX 5

int top=-1,stack[MAX];

void push();

void pop();

void display();

void isEmpty();

void isFull();

void peek();

void reverse();

void iscount();

void search();

void main()

{

int ch;

while(1)

{

printf("\n----------------------------------------------------------------------------\n");

printf("\t1.Push\t2.Pop\t3.Display\t4.isEmpty\t5.isfull\t6.peek\t7.reverse\t8.count\t9.searching\t10.Exit\t");

printf("\n\nEnter your choice:");

scanf("%d",&ch);

switch(ch)

{

case 1: push();

break;

case 2: pop();

break;

case 3: display();

break;

case 4 : isEmpty();

break;

case 5: isFull();

break;

case 6: peek();

break;

case 7 : reverse();

break;

case 8: iscount();

break;

case 9: search();

break;

case 10: exit(0);

break;

default: printf("\nWrong Choice!!");

}

}

}

void push()

{

int val;

if(top==MAX-1)

{

printf("\nStack is full!!");

}

else

{

printf("\nEnter element to push:");

scanf("%d",&val);

top=top+1;

stack[top]=val;

}

}

void pop()

{

if(top==-1)

{

printf("\tStack is empty!!");

}

else

{

printf("\tDeleted element is %d",stack[top]);

top=top-1;

}

}

void display()

{

int i;

if(top==-1)

{

printf("\nStack is empty!!");

}

else

{

printf("\nStack is...\n");

for(i=0;i<=top;i++)

printf("%d\n",stack[i]);

}

}

void isEmpty(){

if(top == -1){

printf("Stack is empty: Underflow State\n");

return 1;

}

printf("Stack is not empty\n");

return 0;

}

void isFull()

{

if(top ==MAX-1)

{

printf("Stack is full:\n");

return 1;

}

printf("Stack Is Not full\n");

return 0;

}

void peek()

{

int x = stack[top];

printf("%d is the top most element of the stack\n", x);

return x;

}

void reverse()

{

int i;

if(top==-1)

printf("Stack is Empty!!!");

else

{

for(i=top;i>=0;i--)

{

printf("\t%d\t",stack[i]);

}

}

}

void iscount()

{

if(top==-1)

printf("Element in stack -: 0");

else

printf("Element in the stack -: %d",(top+1));

}

void search()

{

int x,i,f=0;

printf("Enter the element in the stack to search :\n");

scanf("%d",&x);

i=0;

while(i<=top)

{

if(stack[i]==x)

{

f=1;

break;

}

i++;

}

if(f==1)

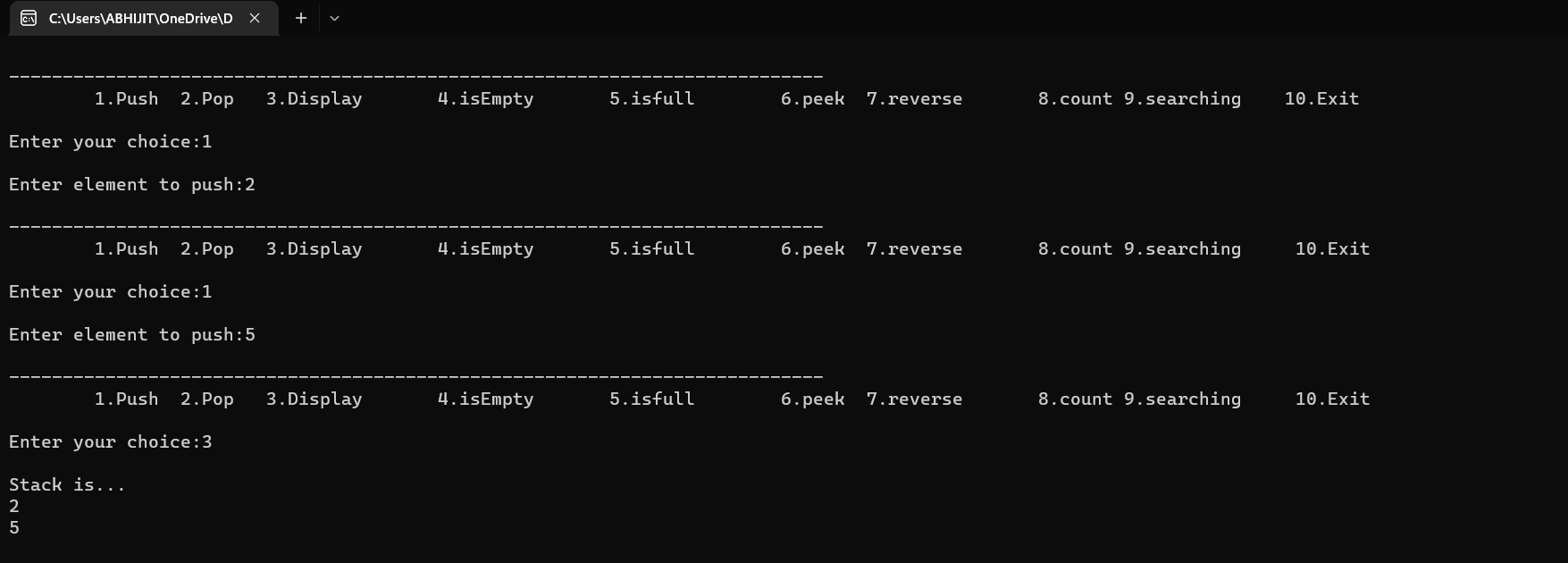
printf("%d\t is found at %d\t th position",x,i);

else

printf("%d is not found ",x);

}

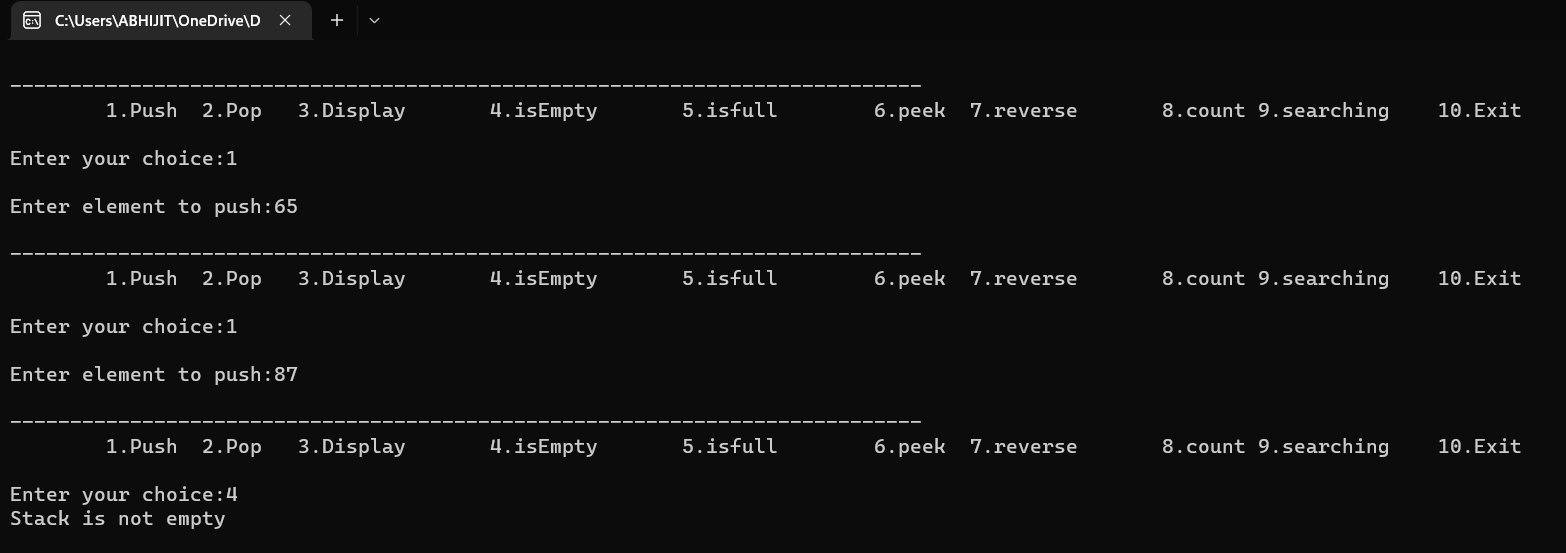
**OUTPUT 1 :** PUSH AND DISPLAY



**OUTPUT 2 :** POP AND DISPLAY



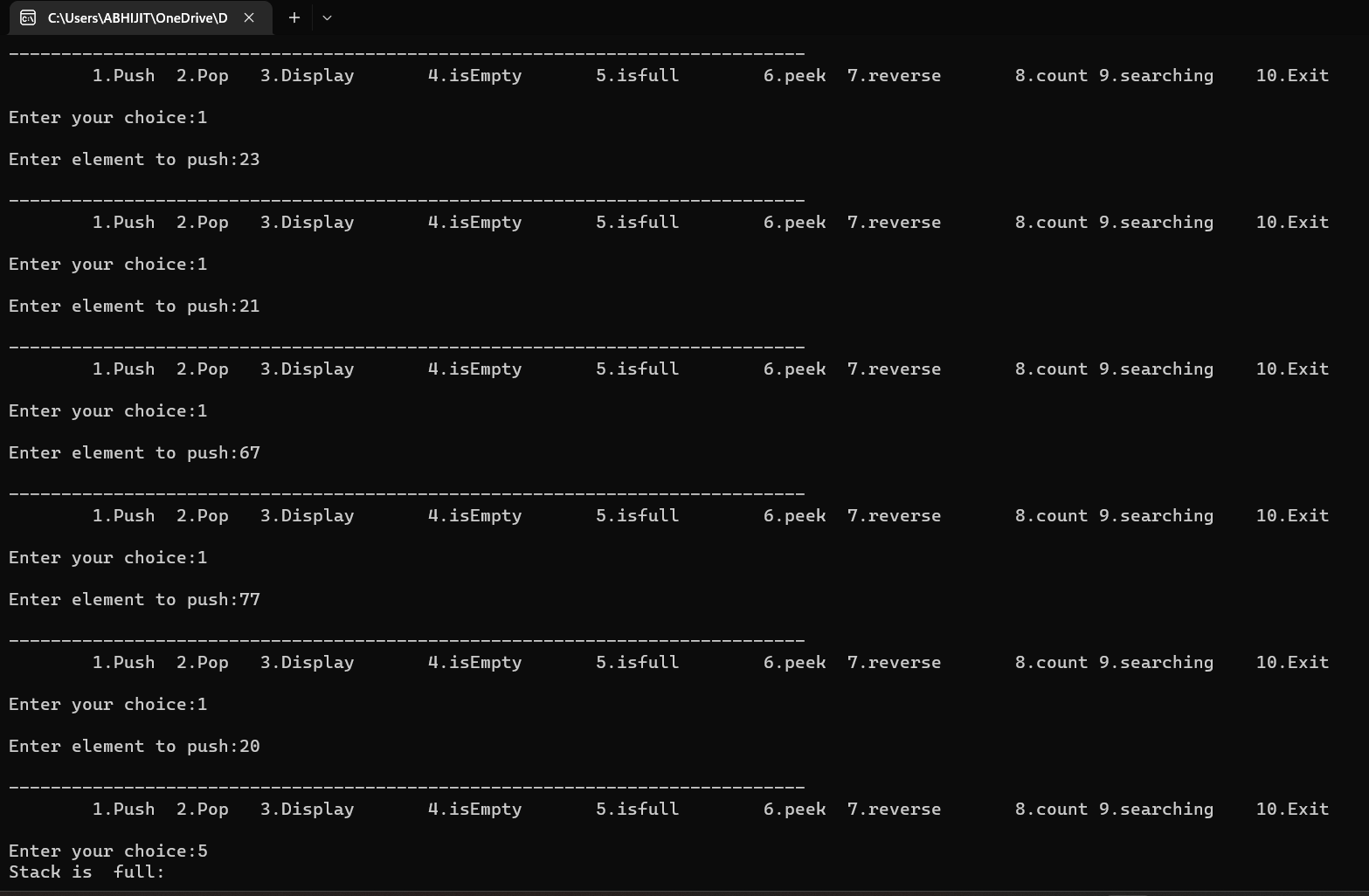
**OUTPUT 3 :** PUSH AND ISEMPTY



**OUTPUT 4 :** PUSH AND ISFULL



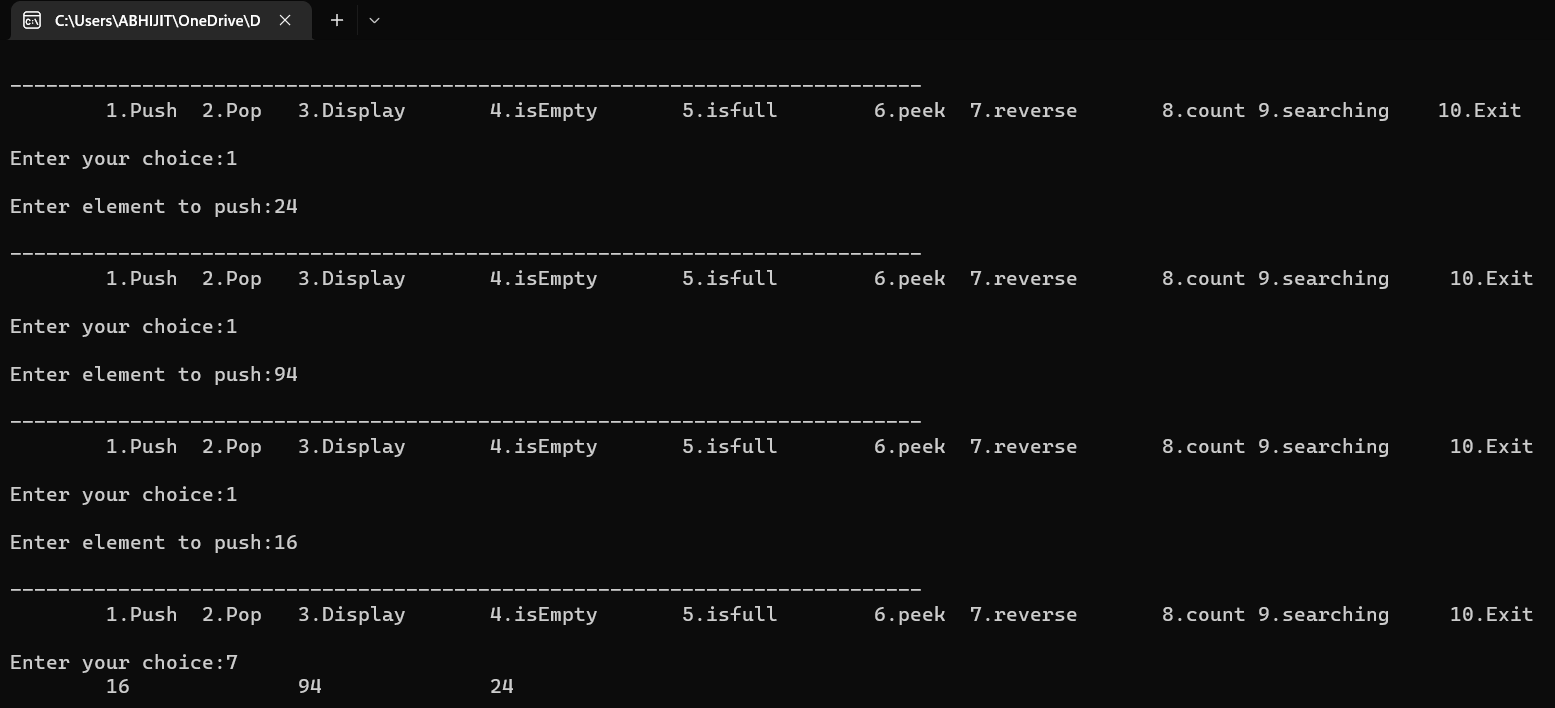
**OUTPUT 5 :** PUSH AND ISFULL



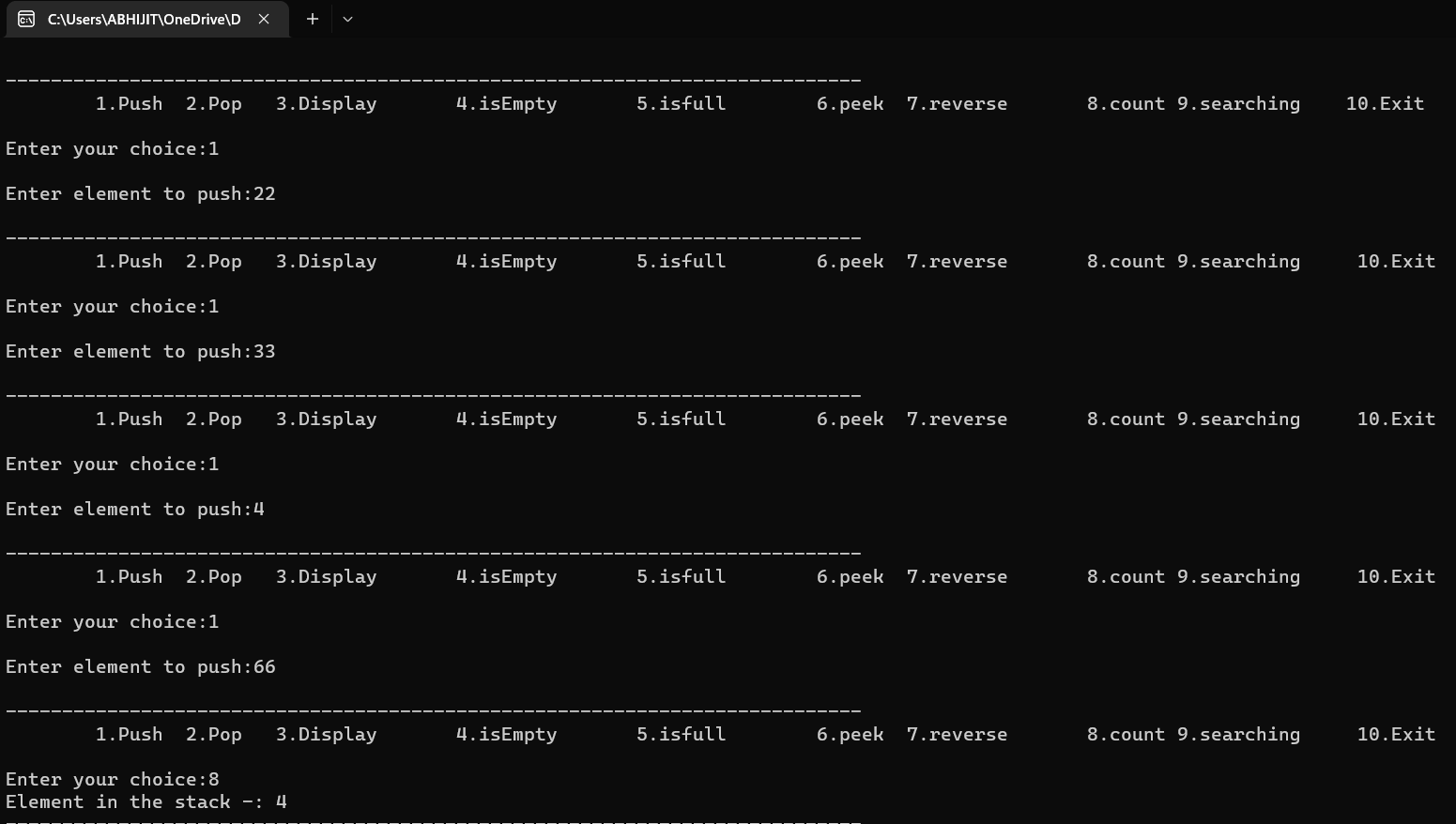
**OUTPUT 6 :** PUSH AND PEEK



**OUTPUT 7 :** PUSH AND REVERSE

****

**OUTPUT 8** : PUSH AND COUNT

****

**OUTPUT 9 :** PUSH AND SEARCHING

****

**OUTPUT 10 :**

