Name: KAMCHE YANN ARNAUD

Matricule : FE21A208

Department: Computer Engineering

Level: 300

Task: Implement a stack using array

1. CODE

```
/*STACK IMPLEMENTATION
Using Arrays
10/20/2022
*/

#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#define MAX_SIZE 100

int A[MAX_SIZE];
int top = -1; // Top iterates through the overall array
int result;

//Create function
void Create(int Size_List)
{
      int A[Size_List];
      return;
}
/*Push function: Inserts an element into the stack
If the stack is full(Overflow), 'Push' will not succeed
*/
```

```
void Push(int max_size, int element)
{
      if( top == max_size - 1)
      {
            printf("Error: STACK OVERFLOW");
            return;
      }

      A[++top] = element; //We use pre-increment operator. Incrementation
will take place first.
}

/*Pop functions removes an element from the array
If the array is empty, an error is displayed
*/
int Pop()
{
      int pop;
      pop = top;
      if( top == -1)
            return 0;
      else
            --top;
                  return A[pop];

}

//sizeOfStack function returns the size of the task
int sizeOfStack()
{
```

```
            int size;
            size = top;
            return ++size;
}

/*TOP fuction returns data at top index*/

int TopOfStack()
{     if(top == -1)
            return 0;
      else
            return A[top];
}

/*Display fuction dispalys the content of the stack*/

void Display()
{
      int i;

      if (top == -1)
            printf("Stack is Empty");
      else
      {
            printf("Stack:  ");
            for(i = 0; i<= top; i++)
                  printf("%d ", A[i]);
            printf("\n");
      }

}
```

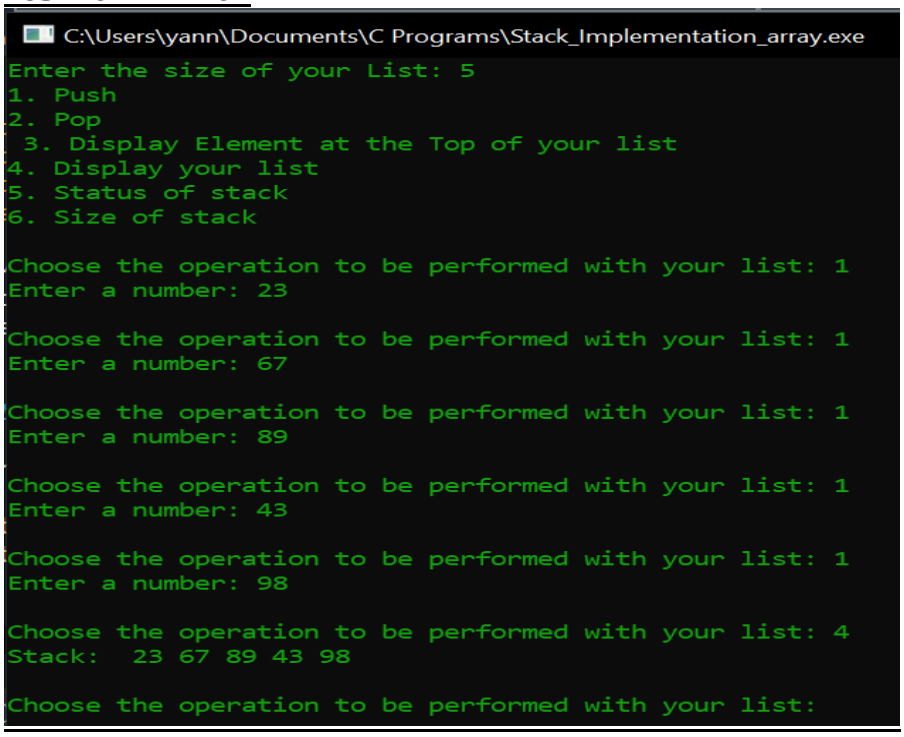
```
//
void Status_Stack(int size)
{
	if(top== -1)
	{
		printf("STATUS: Empty\n");
		return;
	}
	else if (top == size - 1)
	{
		printf("STATUS: FULL\n");
		return;
	}
	else
	{
		printf("STATUS: Not Empty\n");
		return;
	}
}

int main()

{
		system("color 2");
	int choice, num, Size_List;
	printf("Enter the size of your List: ");
	scanf("%d", &Size_List);

	Create(Size_List);
	printf("1. Push\n");
	printf("2. Pop\n ");
```

```
	printf("3. Display Element at the Top of your list\n");
	printf("4. Display your list\n");
	printf("5. Status of stack\n");
	printf("6. Size of stack\n");
	options:
		printf("\nChoose the operation to be performed with your list: ");
	scanf("%d", &choice);

	while(choice == 1|| choice == 2|| choice == 3|| choice == 4 || choice == 5 || choice == 6)
	{

		while(choice==1)
			{
				printf("Enter a number: ");
				scanf("%d", &num);
				Push(Size_List, num);
				goto options;
			}

		while(choice == 2)
		{
			result = Pop();
			if (result == 0)
				printf("Empty");
			else
				printf("%d", result);
			goto options;
		}

		while(choice == 3)
```

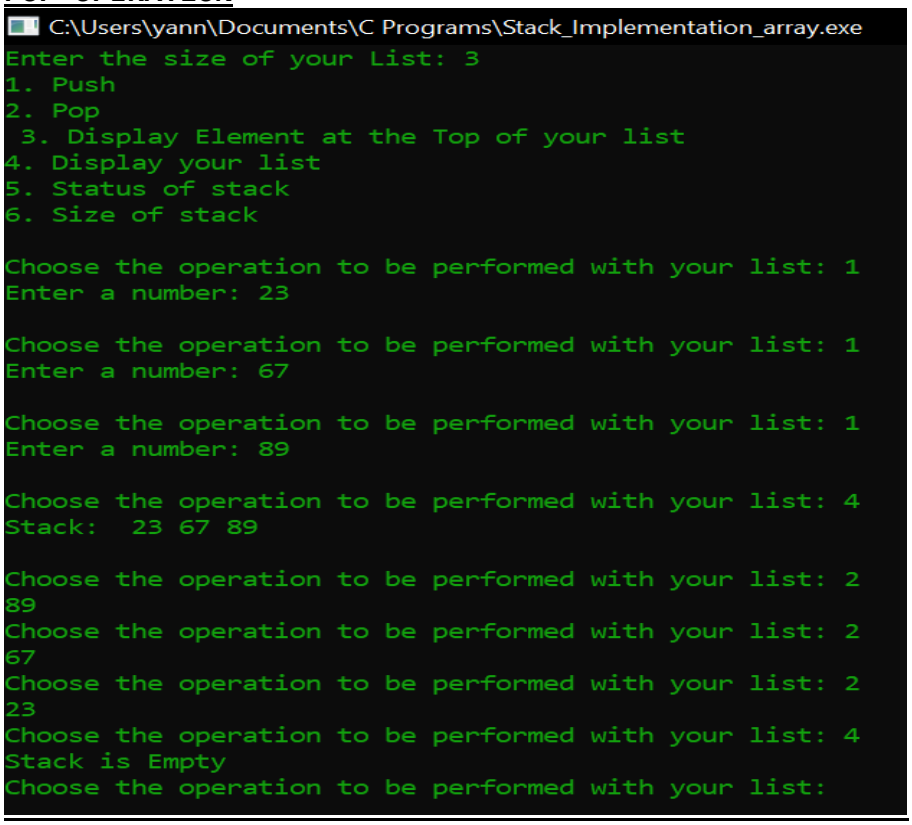
```
		{
			result = TopOfStack();
			if (result == 0)
				printf("Empty");
			else
				printf("%d", TopOfStack());
			goto options;
		}

		while(choice == 4)
		{
			Display(Size_List);
			goto options;
		}
		while(choice == 5){
			Status_Stack(Size_List);
			goto options;
		}
		while(choice == 6){
			result = sizeOfStack();
			if (result == 0)
				printf("Empty");
			else
				printf("%d", sizeOfStack());
			goto options;
		}
	}
	return 0;
}
```

2. COMPILATION RESULTS

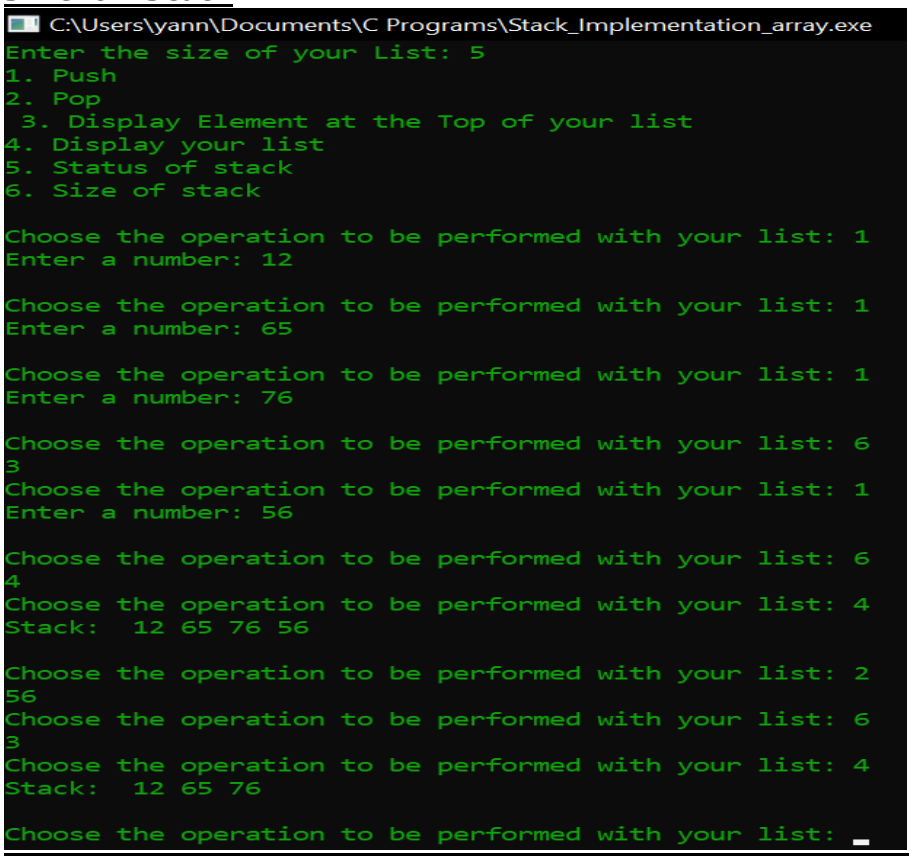
I) PUSH OPERATION



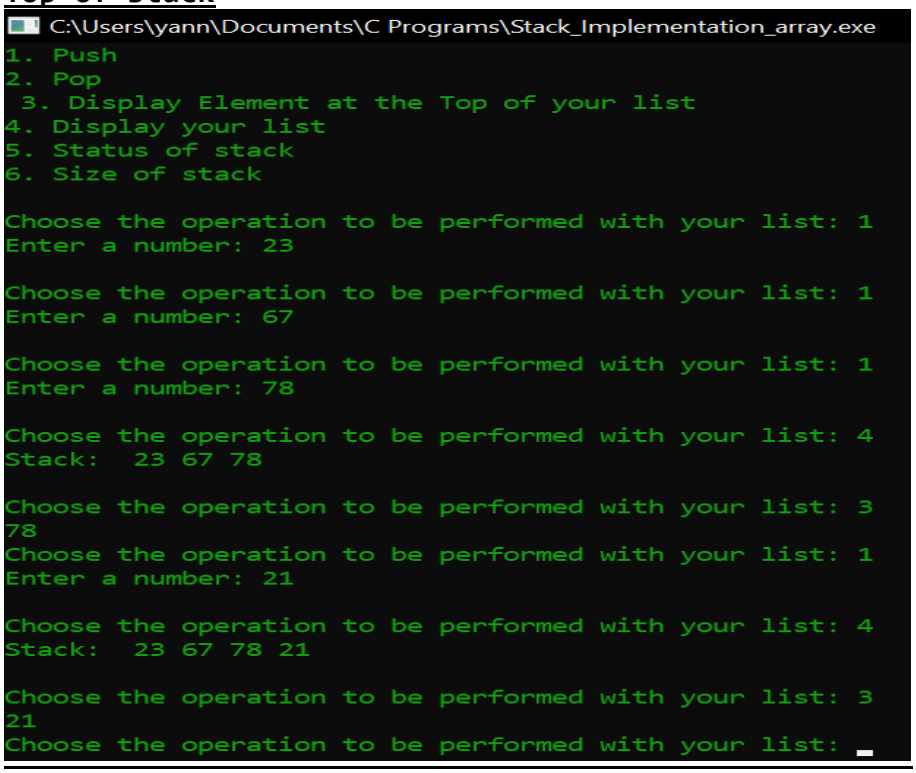
II) POP OPERATION



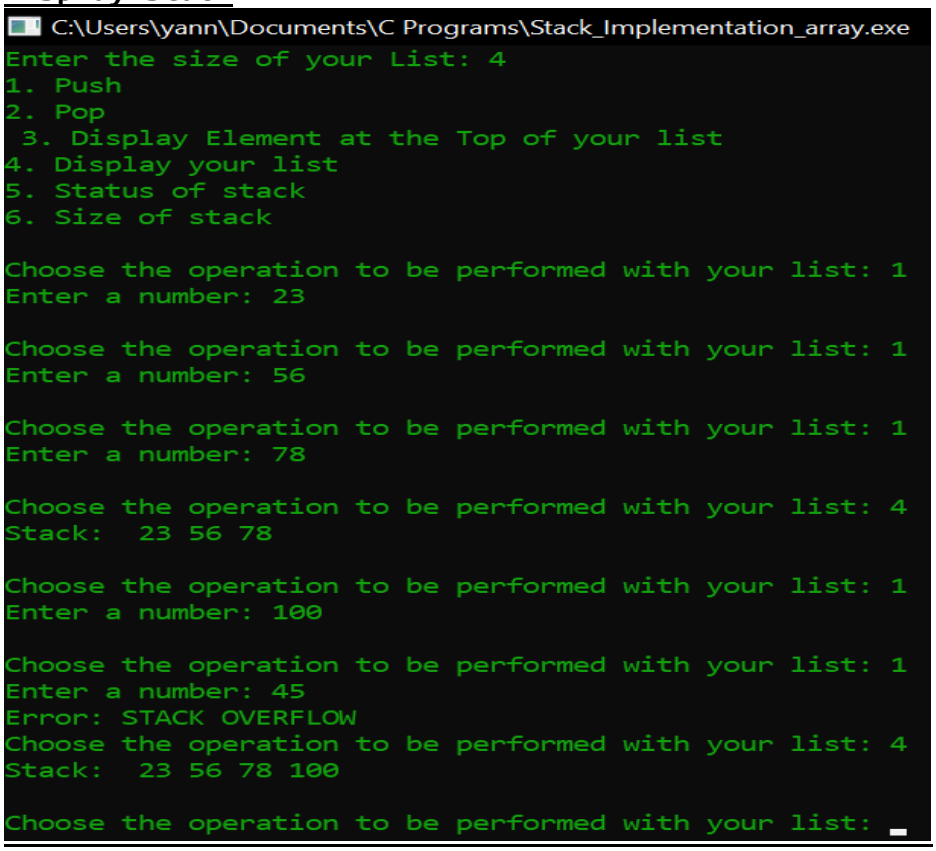
III) Size of stack



IV) Top of stack



V) Display stack



VI) Status of stack