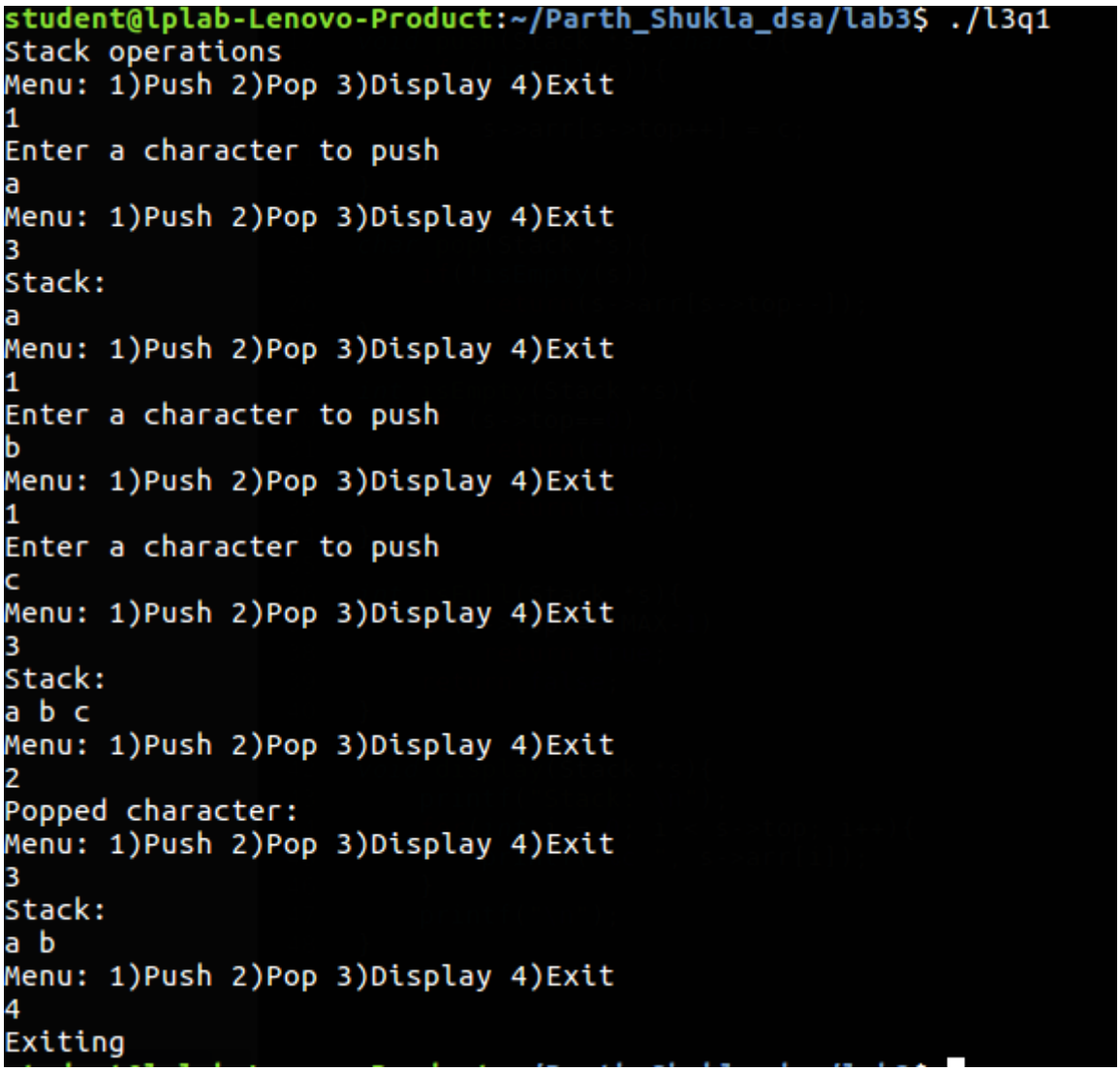
Week3

```
// Stack operations
# define MAX 10
# define true 1
# define false 0

typedef struct{
	char arr[MAX];
	int top;
}Stack;

void push(Stack *s, char c);
char pop(Stack *s);
int isEmpty(Stack *s);
int isFull(Stack *s);
void display(Stack *s);

void push(Stack *s, char c){
	if (!isFull(s)){
		// s->top++;
		s->arr[s->top++] = c;
	}
}

char pop(Stack *s){
	if(!isEmpty(s)){
		s->top--;
		return s->arr[s->top];
		// return(s->arr[s->top--]);
	}
}

int isEmpty(Stack *s){
	if (s->top==-1)
		return(true);
	else
		return(false);
}

int isFull(Stack *s){
	if(s->top == MAX-1)
		return true;
	return false;
}

void display(Stack *s){
	printf("Stack: \n");
	for(int i = 0; i < s->top; i++){
		printf("%c ", s->arr[i]);
	}
	printf("\n");
```

}
1)
// Implement a menu driven program to define a stack of characters. Include push, pop and
// display functions. Also include functions for checking error conditions such as underflow
// and overflow (ref. figure 1) by defining isEmpty and isFull functions. Use these function
// in push, pop and display functions appropriately. Use type defined structure to define a
// STACK containing a character array and an integer top. Do not use global variables.

```
#include <stdio.h>
#include <stdlib.h>
#include "stack_operations.h"

int main(){
	printf("Stack operations\n");
	Stack s;
	int ch = 0;
	do{
		ch = 0;
		printf("Menu: 1)Push 2)Pop 3)Display 4)Exit\n");
		scanf("%d", &ch);
		// printf("dhwj\n");
		switch(ch){
			case 1:
			printf("Enter a character to push\n");
			char c;
			scanf(" %c", &c);
			push(&s, c);
			break;
			case 2:
			printf("Popped character: ");
			char p = pop(&s);
			printf("%c\n", p);
			break;
			case 3:
			display(&s);
			break;
			case 4:
			printf("Exiting\n");
			break;
			default: printf("Invalid operation\n");
		}
	}while(ch != 4);
	return 0;
}
```



2)
// Convert a given decimal number to binary using stack.

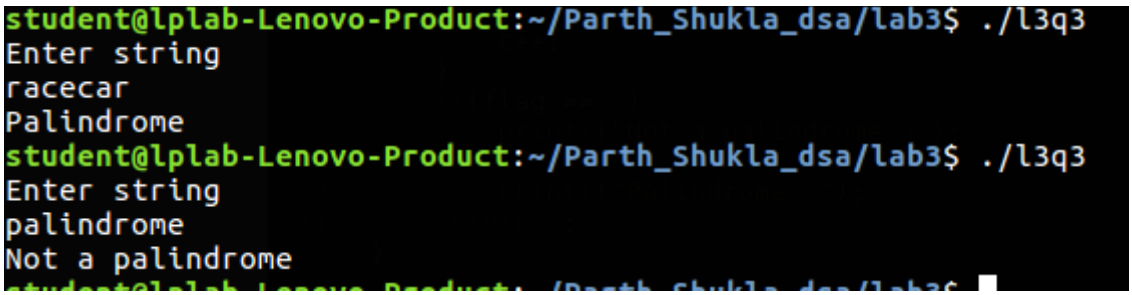
```
#include <stdio.h>
#include <stdlib.h>
#include "stack_operations.h"

int main(){
	int dec;
	printf("Enter decimal: ");
	scanf("%d", &dec);
	if(dec > 511){
		printf("Number too big\n");
	}
	else{
		Stack s;
		while(dec != 0){
			push(&s, dec%2+'0');
			dec = dec/2;
		}
		while(isEmpty(&s) == 0){
			printf("%c", pop(&s));
```

```
        }
        printf("\n");
    }
}
```

```
student@lplab-Lenovo-Product:~/Parth_Shukla_dsa/lab3$ ./l3q2
Enter decimal: 8
1000
student@lplab-Lenovo-Product:~/Parth_Shukla_dsa/lab3$ ./l3q2
Enter decimal: 27
11011
student@lplab-Lenovo-Product:~/Parth_Shukla_dsa/lab3$ ./l3q2
Enter decimal: 156
◆10011100
```

3)
// Determine whether a given string is palindrome or not using stack.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "stack_operations.h"

int main(){
    char *str = (char *)calloc(5, sizeof(char));
    printf("Enter string\n");
    scanf("%s", str);

    Stack s;
    for(int i = 0; i < strlen(str); i++){
        push(&s, str[i]);
        // display(&s);
    }
    int c = 0, flag = 0;
    while(s.top > c){
        char e = pop(&s);
        // printf("%c\n", e);
        if(str[c] != e){
            flag = 1; break;
        }
        c++;
    }
    if(flag == 1)
        printf("Not a palindrome\n");
    else
        printf("Palindrome\n");
    return 0;
}
```


student@lplab-Lenovo-Product:~/Parth_Shukla_dsa/lab3$ ./l3q3
Enter string
racecar
Palindrome
student@lplab-Lenovo-Product:~/Parth_Shukla_dsa/lab3$ ./l3q3
Enter string
palindrome
Not a palindrome