stack-using-array.c

```
//write a program to implement stack in c using array.

#include <stdio.h>
#include <stdlib.h>
#define max 10

void push(int x);
int pop();
void display();
int isEmpty();
int isFull();
int len();
int topItem();

int stack[max];
int top = -1;

int main(){
   int op,x;
   while(1){
      printf("1.Push, 2.Pop, 3.Display, 4. Length, 5. isEmpty, 6. isFull, 7. Top, 8.Exit.\nEnter Your choice:
");
      scanf("%d",&op);
      switch(op){
         case 1:
            if(isFull()){
               printf("Overflow.\n");
            }else{
               printf("Enter the item: ");
               scanf("%d",&x);
               push(x);
            }
            break;
         case 2:
            if(isEmpty()){
               printf("Underflow.\n");
            }else{
               x = pop();
               printf("Removed %d from stack\n",x);
            }
            break;
         case 3:
            if(isEmpty()){
               printf("Stack is empty.\n");
            }else{
               printf("The elements of stack are: ");
               display();
            }
            break;
         case 4:
            printf("%d\n",len());
            break;
         case 5:
            if(isEmpty()){
```

```
            printf("Stack is Empty\n");
         }else{
            printf("Stack is Not Empty\n");
         }
         break;
      case 6:
         if(isFull()){
            printf("Stack is Full\n");
         }else{
            printf("Stack is Not Full\n");
         }
         break;
      case 7:
         if(isEmpty()){
            printf("Stack is Empty\n");
         }else{
            printf("%d\n",topItem());
         }
         break;
      case 8:
         exit(0);
      default:
         printf("Invalid Input.\n");
    }
  }
  return 0;
}

void push(int x){
  stack[++top] = x;
}

int pop(){
  return stack[top--];
}

void display(){
  for(int i = 0; i < top;i++){
    printf("%d -> ",stack[i]);
  }
  printf("%d\n",stack[top]);
}
int isEmpty(){
  if(top == -1){
    return 1;
  }
  return 0;
}
int isFull(){
  if(top == max-1){
    return 1;
  }
  return 0;
}
int len(){
  return top+1;
}
```

```
int topItem(){
   return stack[top];
}
```

OUTPUT

```
PS S:\WorkSpace\CollegeWork\DataStructure> gcc .\stack-using-array.c
PS S:\WorkSpace\CollegeWork\DataStructure> ./a
1.Push, 2.Pop, 3.Display, 4. Length, 5. isEmpty, 6. isFull, 7. Top, 8.Exit.
Enter Your choice: 1 12 1 13 1 14 1 115 1 17 3
The elements of stack are: 12 -> 13 -> 14 -> 115 -> 17
1.Push, 2.Pop, 3.Display, 4. Length, 5. isEmpty, 6. isFull, 7. Top, 8.Exit.
Enter Your choice: 3
The elements of stack are: 12 -> 13 -> 14 -> 115 -> 17
1.Push, 2.Pop, 3.Display, 4. Length, 5. isEmpty, 6. isFull, 7. Top, 8.Exit.
Enter Your choice: 2
Removed 17 from stack
1.Push, 2.Pop, 3.Display, 4. Length, 5. isEmpty, 6. isFull, 7. Top, 8.Exit.
Enter Your choice: 3
The elements of stack are: 12 -> 13 -> 14 -> 115
1.Push, 2.Pop, 3.Display, 4. Length, 5. isEmpty, 6. isFull, 7. Top, 8.Exit.
Enter Your choice: 4
4
1.Push, 2.Pop, 3.Display, 4. Length, 5. isEmpty, 6. isFull, 7. Top, 8.Exit.
Enter Your choice: 7
115
1.Push, 2.Pop, 3.Display, 4. Length, 5. isEmpty, 6. isFull, 7. Top, 8.Exit.
Enter Your choice: 8
PS S:\WorkSpace\CollegeWork\DataStructure>
```