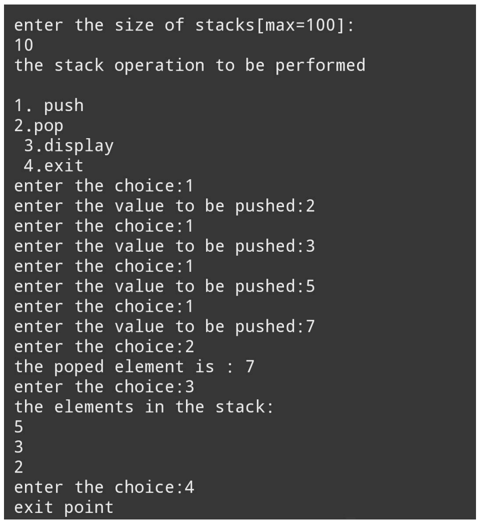
LAB TASK_4

```
#include<stdio.h>
int stack[100], choice, n, top, x;
void push(void);
void pop(void);
void display(void);

int main() {
   top = -1;
   printf("enter the size of stacks[max=100]:\n");
   scanf("%d", &n);
   printf("the stack operation to be performed\n");
   printf("\n1. push\n2.pop\n 3.display\n 4.exit\n");
   do {
      printf("enter the choice:");
      scanf("%d", &choice);
      switch (choice) {
      case 1:
         push();
         break;
      case 2:
         pop();
         break;
      case 3:
         display();
         break;
      case 4:
         printf("exit point");
         break;
      default:
         printf("\n enter the valide choice\n");
      }
   } while (choice != 4);
   return 0;
}

void push() {
   if (top == n - 1) {
      printf("\nstack is overflow");
   } else {
      printf("enter the value to be pushed:");
      scanf("%d", &x);
```

```
        top++;
        stack[top] = x;
    }
}
void pop() {
    if (top == -1) {
        printf("stack is underflow");
    } else {
        printf("the poped element is : %d\n", stack[top]);
        top--;
    }
}
void display() {
    int i;
    if (top >= 0) {
        printf("the elements in the stack:\n");
        for (i = top; i >= 0; i--)
            printf("%d\n", stack[i]);
    } else {
        printf("the stack is empty ");
    }
}
```

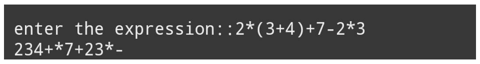


```
enter the size of stacks[max=100]:
10
the stack operation to be performed

1. push
2.pop
 3.display
 4.exit
enter the choice:1
enter the value to be pushed:2
enter the choice:1
enter the value to be pushed:3
enter the choice:1
enter the value to be pushed:5
enter the choice:1
enter the value to be pushed:7
enter the choice:2
the poped element is : 7
enter the choice:3
the elements in the stack:
5
3
2
enter the choice:4
exit point
```

```
#include<stdio.h>
#include<ctype.h>
char stack[100];
int top=-1;
void push(char x)
{
    stack[top++]=x;
```

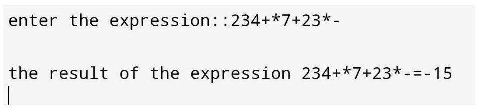
```
}
char pop()
{
if(top==-1)
return -1;
else
  return stack[top–];
int priority(char x)
{
   if (x == '(')
       return 0;
   if (x == '+' || x == '-')
       return 1;
   if (x == '*' || x == '/')
       return 2;
   if (x == '^')
       return 3;
   return -1;
}

int main()
{
   char exp[100];
   char *e,x;
   printf("enter the expression::");
   scanf("%s", exp);
   e=exp;
   while (*e != '\0')
   {
       if (isalnum(*e))
           printf("%c", *e);
       else if (*e == '(')
           push(*e);
       else if (*e == ')')
       {
           while ((x = pop()) != '(')
               printf("%c", x);
       }
       else
       {
           while (priority(Stack[top]) >= priority(*e))
               printf("%c", pop());
           push(*e);
       }
```

```
        e++;
    }
    while (top != -1)
    {
        printf("%c", pop());
    }
    return 0;
}
```



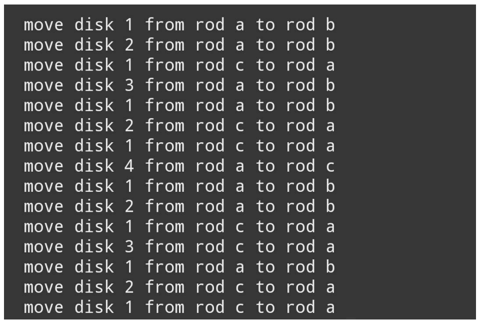
```
enter the expression::2*(3+4)+7-2*3
234+*7+23*-
```

```
#include<stdio.h>
#include<ctype.h>
char stack[20];
int top=-1;
void push(char x)
{
    stack[++top]=x;
}
int pop()
{
   return stack[top--];
   }
   int main()
   {
   char exp[20];
   char *e;
   int n1,n2,n3,num;
   printf("enter the expression::");
   scanf("%s",exp);
   e=exp;
   while(*e!='\0')
   {
   if(isdigit(*e))
   {
   num=*e-48;
   push(num);
   }
   else
   {
   n1=pop();
   n2=pop();
   switch(*e)
   {
   case '+':
   {
   n3=n1+n2;
   break;
   }
  case '-':
   {
   n3=n1-n2;
   break;
   }
  case '*':
   {
   n3=n1*n2;
   break;
   }
  case '/':
   {
   n3=n2/n1;
   break;
   }
 }
push(n3);
   }
  e++;
   }
   printf("\nthe result of the expression %s=%d\n", exp , pop());
   return 0;
   }
```



```
enter the expression::234+*7+23*-

the result of the expression 234+*7+23*-=-15
```

```
#include<stdio.h>

void toh(int n, char a, char c, char b)
{
    if(n==1)
    {
        printf("\n move disk 1 from rod %c to rod %c ", a,b);
        return;
    }
    toh(n-1, 'a', 'c', 'b'); // move n-1 disks from 'a' to 'c', using 'b' as auxiliary peg
    printf("\n move disk %d from rod %c to rod %c", n,a,b);
    toh(n-1, 'c', 'b', 'a'); // move n-1 disks from 'c' to 'b', using 'a' as auxiliary peg
}

int main()
{
    int n=4;
    toh(n,'a','b','c');
    return 0;
}
```



```
move disk 1 from rod a to rod b
move disk 2 from rod a to rod b
move disk 1 from rod c to rod a
move disk 3 from rod a to rod b
move disk 1 from rod a to rod b
move disk 2 from rod c to rod a
move disk 1 from rod c to rod a
move disk 4 from rod a to rod c
move disk 1 from rod a to rod b
move disk 2 from rod a to rod b
move disk 1 from rod c to rod a
move disk 3 from rod c to rod a
move disk 1 from rod a to rod b
move disk 2 from rod c to rod a
move disk 1 from rod c to rod a
```