

Practica 2

Estructura de Datos

1. Introducción

Las pilas son estructuras de datos que tienen una característica, los elementos de la pila o stack se agregan y se sacan desde el tope de la misma y solo desde el tope, el programa que se desarrolla tiene como objetivo leer línea por línea, cada línea esta comprendida una expresión aritmética la cual se debe convertir de infijo a posfijo y a su vez ser evaluada empleando en ambos procesos una pila. Es decir, creando una apariencia de calculadora. La expresión prefija nos indica que el operador va antes de los operandos sus características principales son:

- Los operandos conservan el mismo orden que la notación infija equivalente.
- No requiere de paréntesis para indicar el orden de precedencia de operadores ya que el es una operación.
- Se evalúa de izquierda a derecha hasta que encontremosle primer operador seguido inmediatamente de un par de operandos.
- Se evalúa la expresión binaria y el resultado se cambia como un nuevo operando. Se repite este hasta que nos quede un solo resultado.

Las expresiones posfija, Como su nombre lo indica se refiere a que el operador ocupa la posición después de los operandos sus características principales son:

- -El orden de los operandos se conserva igual que la expresión infija equivalente no utiliza paréntesis ya que no es una operación ambigua
- -La operación posfija no es exactamente lo inverso a la operación prefija equivalente.

2. Código

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  #include <ctype.h>
5  #include "pila.h"
6
7  int precedencia(char opc){
8
9      int pre;
10
11     if(opc == '+' || opc == '-')
12         pre = 1;
13     else if(opc == '*' || opc == '/')
14         pre = 2;
15     else if(opc == '^')
16         pre = 3;
17     else
18         pre = 4;
19
20     return pre;
21 }
22
23 void verificar_operador(pila* pila, Info inf, int *cont){
24
```

```

25  int pre_act,pre_pil,ncont;
26  Info inf_aux;
27  char aux;
28
29  if (empty(*pila))
30      push(pila,inf);
31
32  else{
33      inf_aux = top(*pila);
34      pre_act = precedencia(inf.caracter);
35      pre_pil = precedencia(inf_aux.caracter);
36
37      if(pre_act == 4)
38          {
39              while (!empty(*pila))
40                  {
41                      printf("%c",inf_aux.caracter);
42                      pop(pila);
43                      if(empty(*pila))
44                          break;
45                      else{
46                          inf_aux = top(*pila);
47                          printf("%c",inf_aux.caracter);
48                          pop(pila);
49                      }
50                  }
51
52      }else if (pre_act == pre_pil)
53      {
54          printf("%c",inf_aux.caracter);
55          pop(pila);
56          push(pila,inf);
57          inf_aux = top(*pila);
58
59      }else if(pre_act > pre_pil){
60          push(pila,inf);
61      }
62      else if(pre_act < pre_pil){
63          while (!empty(*pila))
64              {
65                  printf("%c",inf_aux.caracter);
66                  pop(pila);
67                  if(empty(*pila))
68                      break;
69                  else{
70                      inf_aux = top(*pila);
71                      pop(pila);
72                  }
73              }
74
75          push(pila, inf);
76      }
77      else
78          printf("Incorrecto\n");
79
80  }
81  }
82
83  void cambiar_postfijo(char* op){
84      char carac = ' ';
85      int cont = 0,i=0;

```

```

86  pila P;
87  Info inf;
88
89  crearpila(&P);
90
91  do
92  {
93      carac = op[i];
94      if(isdigit(carac)){
95          printf("%c",carac);
96          if(cont == 2){
97              cont = 0;
98              inf = top(P);
99              pop(&P);
100             if(empty(P))
101                 ++cont;
102
103         }
104     }
105     else{
106         inf.character = carac;
107         verificar_operador(&P,inf,&cont);
108     }
109     ++i;
110 }while(carac!= '\n');
111 }
112
113 int main(int argc, char *argv[]) {
114
115     FILE* operaciones = fopen("Operaciones.txt","r+");
116     char* operacion;
117
118     do{
119         fgets(operacion,20,operaciones);
120         printf("\nOperacion: %s",operacion);
121         printf("Notacion postfija:");
122         cambiar_postfijo(operacion);
123         printf("\n");
124     }while(!feof(operaciones));
125
126     fclose(operaciones);
127
128     return 0;
129 }

```

3. Extensión de Funcionalidades

- Podríamos crear una calculadora más compleja que permita evaluar integrales y derivadas.
- Podríamos realizar una visualización paso a paso para poder entender de mejor manera como se mueven las notaciones.

4. Programas de se emplea una pila

- Gestión de ventanas en Windows o Linux (cuando cerramos una ventana siempre recuperamos la que teníamos detrás).
- Editores de texto u otras herramientas, donde el usuario puede deshacer los cambios mediante la operación "deshacer", la cual extrae el estado del texto o cualquier elemento, antes del último cambio realizado.

5. Capturas de Pantalla

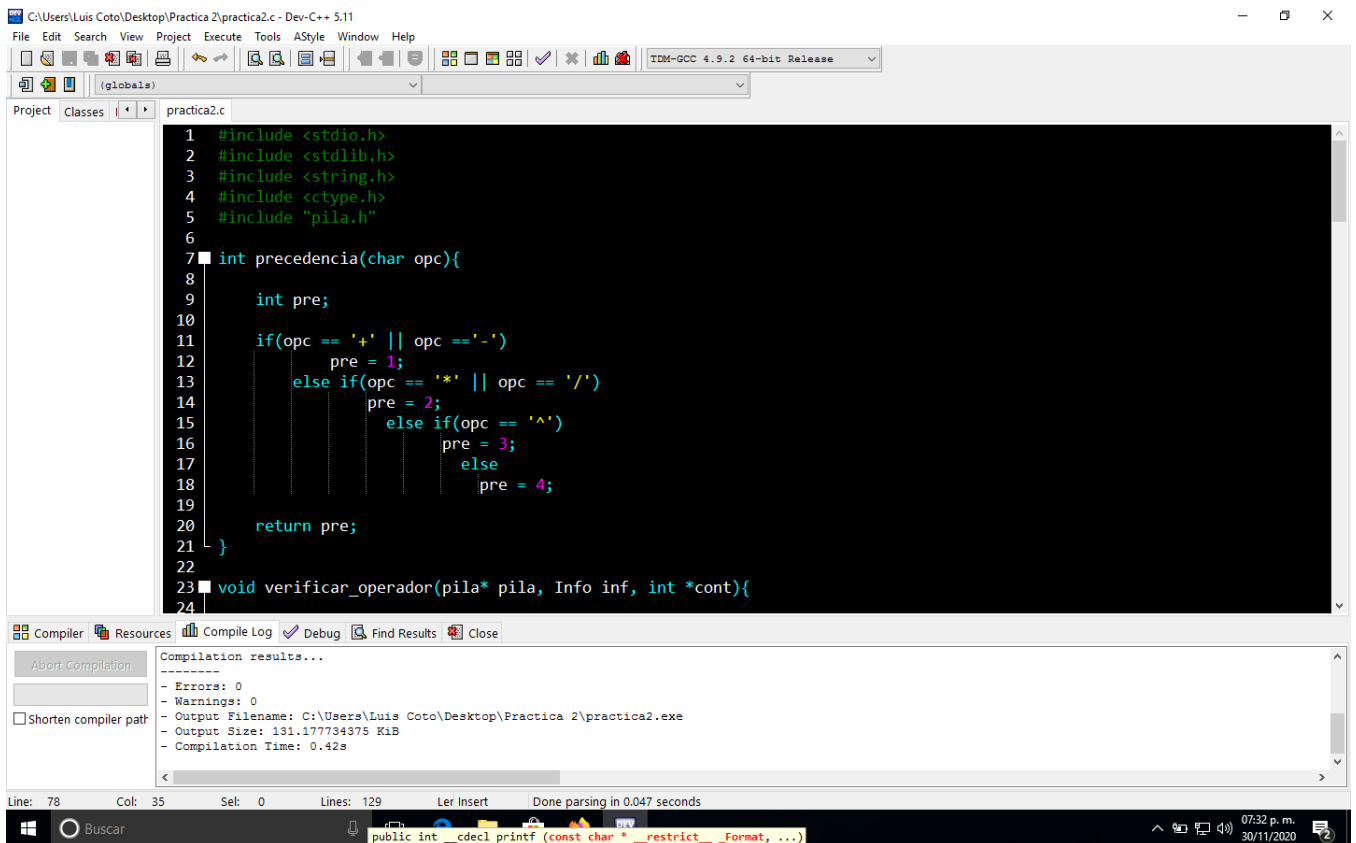


Figura 1: Pantalla de DevC

```
Compilation results...
-----
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\Luis Coto\Desktop\Practica 2\practica2.exe
- Output Size: 131.177734375 KiB
- Compilation Time: 0.42s
```

Figura 2: Diagnostico de ejecución

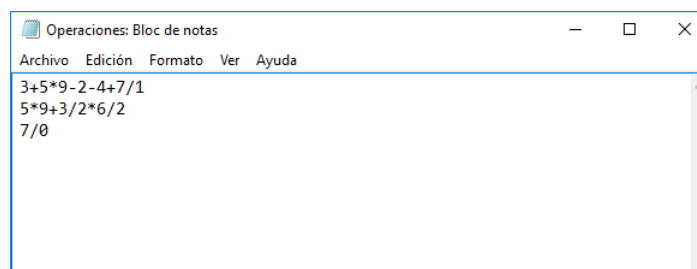


Figura 3: Archivo de operaciones

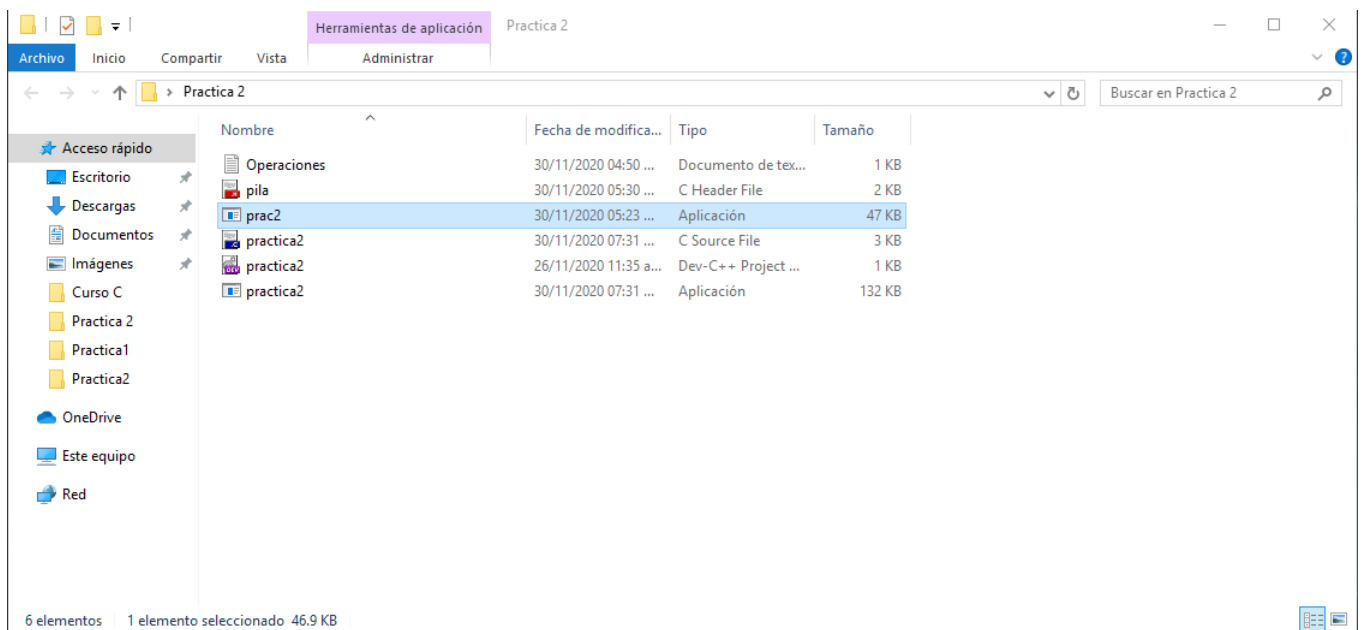


Figura 4: Carpeta de localización

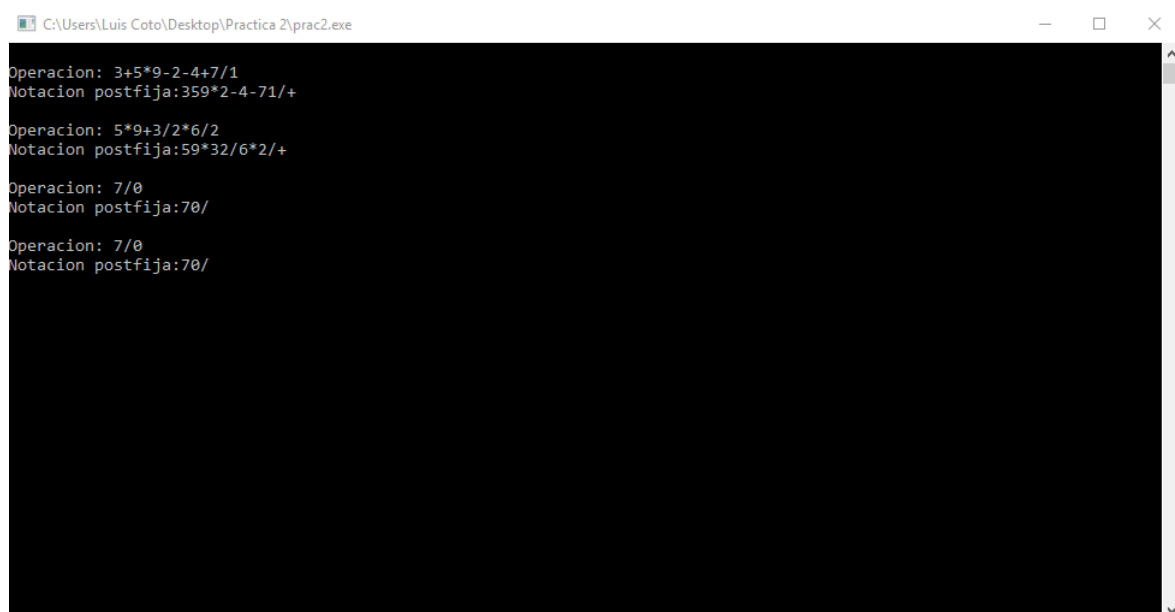


Figura 5: Ejecución del programa