



INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

PRÁCTICA IV: TAD PILA-BALANCEO DE PARÉNTESIS

ELÍAS LÓPEZ RIVERA

2CV5

ALGORITMOS Y ESTRUCTURAS DE DATOS

MAESTRO: MANUEL PORTILLO

Fecha de entrega: 21/05/2025



Algoritmo de conversión Polaca

Algoritmo

TAD pila

Estructura de datos tipo LIFO, es decir el último que entra es el primero en salir, su funcionamiento emula el de una pila de libros, es bastante utilizada para simular entornos dentro de aplicaciones web y en específico será usada para la realización de este algoritmo

Chequeo de paréntesis

Paso 1: Iteramos sobre las posiciones de nuestra cadena

Paso 2: Si el valor de la posición de la cadena es igual a un paréntesis abierto apilamos

Paso 3: Si este corresponde a un paréntesis de cierre despilamos, si desapilamos una pila vacía la expresión está mal escrita

Paso 4: Si al final del proceso la pila está vacía la expresión es correcta, si tiene algún elemento la expresión está mal escrita

Pseudocodigo

Algorithm 1 Conversión Polaca

Require: $[1 + 2 * 7 + (2 + 1) * 9]$

Ensure: *expresincorrecta*

INICIO

char temp=+; **Desde** $i = 1$ **hasta** $i < tamarray$

Si A[i] es (

ApilarA[i]

Si A[i] es)

Desapilar

si **Desapilar**=error

: **imprimir** expresión no valida

Si la pila esta vacia

Imprimir expresión valida

: **Si no**

Imprimir expresión invalida

FIN

Implementación con pila dinámica

```
1 #include <stdio.h>
2 #include <malloc.h>
3 #include <stdbool.h>
4 #include <string.h>
5
6 struct nodo
7 {
8     char ele;
9     struct nodo *siguiente;
10 };
11
12 struct nodo *tope=NULL;
13
14 int isempty()
15 {
16     if(tope==NULL)
17     {
18         return 1;
19     }
20     return 0;
21 }
22
23 void imprimirarreglo(char c[13])
24 {
25     for (int l = 0; l < 13; l++)
26     {
27         printf("%c ",*(c+l));
28     }
29     printf("\n");
30 }
31
32 void push(char val)
33 {
34     struct nodo* nuevo=(struct nodo*)malloc(sizeof(struct nodo));
35     nuevo->ele=val;
36     nuevo->siguiente=tope;
37     tope=nuevo;
38 }
39
40 int pop()
41 {
42     if(isempty())
43     {
44         //printf("ERROR");
45         return 0;
46     }
47     else
48     {
49         struct nodo*temp=tope;
50         if((temp->ele!='(')&(temp->ele!=')'))
51         {
52             printf("%c ",temp->ele);
```

```

53     }
54     tope=temp->siguiente;
55     //char z=temp->ele;
56     free(temp);
57     return 1;
58 }
59 }
60
61
62 int parenthesis(char C[100])
63 {
64     while(*(C++))/*mientras el valor de la
65     isesima posicion sea diferenete de NULL*/
66
67     {
68         if (*C=='(')
69             /*si lo apuntado es un parentesis
70             de abertura apilamos*/
71             {
72                 push(*C);
73             }
74         if (*C==')')
75             /*si lo apuntado es un parentesis
76             de cierre desapilamos
77             */
78             {
79                 if(!pop())
80                     /*si dessapilamos una pila
81                     vacia retornamos 0*/
82                     {
83                         return 0;
84                     }
85             }
86     }
87     if (isempty())
88     {
89         return 1;
90         /*si la pila esta vacia
91         retornamos 1*/
92     }
93
94     return 0;
95     //si no retornamos 0
96 }
97 int main()
98 {
99     char A[100];
100    fgets(A,100,stdin);
101    if( parenthesis(A))
102        /*si la valuacion de parentesis
103        fue positiva o negativa imprimimos
104        el respectivo mensaje*/
105    {
106        printf("Expresion correcta");

```

```

107     }
108     else
109     {
110         printf("Expresion incorrecta");
111     }
112 }

```

Compilación

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
● * Executing task: C:/windows/System32/cmd.exe /d /c .\build\Debug\outDebug.exe

1+2*(3+2)(+)
Expresion correcta
Terminal will be reused by tasks, press any key to close it.

```

Implementación con pila estática

```

1  #include <stdio.h>
2  #include <stdbool.h>
3  #include <string.h>
4  #define MAX 100
5
6  void imprimirarreglo(char c[13])
7  {
8      for (int l = 0; l < 13; l++)
9      {
10         printf("%c ",*(c+l));
11     }
12     printf("\n");
13 }
14
15 struct pila
16 {
17     char ele[MAX];
18     int tope;
19 };
20
21 void incializar(struct pila *p)
22 {
23     p->tope=-1;
24 }
25
26 int isempty(struct pila *p)

```

```

27 {
28     if(p->tope==-1)
29     {
30         return 1;
31     }
32     return 0;
33 }
34
35 bool isfull(struct pila *p)
36 {
37     if(p->tope==MAX-1)
38     {
39         return true;
40     }
41     return false;
42 }
43
44
45 void push(struct pila *p, char val)
46 {
47     if(isfull(p))
48     {
49         printf("La pila esta llena");
50     }
51     else
52     {
53         p->ele[++(p->tope)]=val;
54     }
55 }
56
57 int pop(struct pila *p)
58 {
59     if(isempty(p))
60     {
61         return 0;
62     }
63     else
64     {
65         if((p->ele[p->tope]!='(')&(p->ele[p->tope]!=')'))
66         {
67             printf("%c ",p->ele[p->tope]);
68         }
69         //char z=p->ele[p->tope];
70         (p->tope)--;
71         return 1;
72     }
73 }
74
75
76 int parenthesis(char C[100], struct pila *p)
77 {
78     while(*(C++))
79     {
80         if (*C=='(')

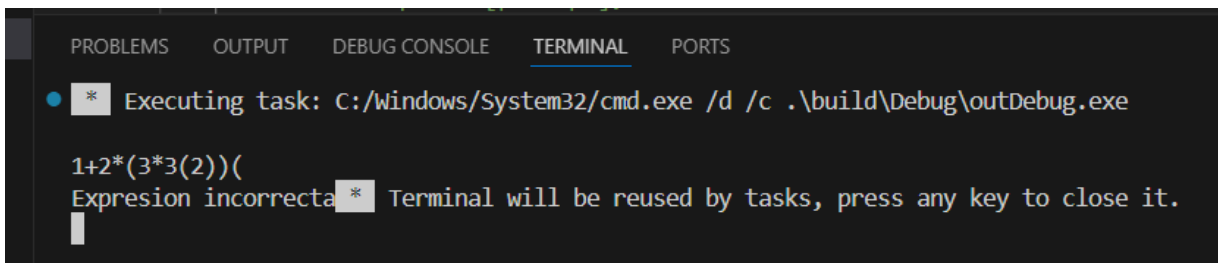
```

```

81     {
82         push(p,*C);
83     }
84     if (*C==' ')
85     {
86         if (!pop(p))
87         {
88             return 0;
89         }
90     }
91 }
92 if (isempty(p))
93 {
94     return 1;
95 }
96
97 return 0;
98 }
99
100 int main()
101 {
102     struct pila p;
103     incializar(&p);
104     char A[100];
105     fgets(A,100,stdin);
106     if( parentesis(A,&p))
107     {
108         printf("Expresion correcta");
109     }
110     else
111     {
112         printf("Expresion incorrecta");
113     }
114 }

```

Compliación



```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
● * Executing task: C:/Windows/System32/cmd.exe /d /c .\build\Debug\outDebug.exe

1+2*(3*3(2)) (
Expresion incorrecta * Terminal will be reused by tasks, press any key to close it.

```


Conclusión

A través de la implementación de una estructura de datos logramos implementar el algoritmo de revisión de paréntesis, lo cual nos mostro una nueva manera de implementar las funciones nativas de una pila