

INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

PRÁCTICA III: TAD PILA

ELÍAS LÓPEZ RIVERA

2CV5

ALGORITMOS Y ESTRUCTURAS DE DATOS

MAESTRO: MANUEL PORTILLO

Fecha de entrega: 05/05/2025



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

Conversión polaca

Paso 1: Iteramos sobre las posiciones de nuestro arreglo char

Paso 2: Si el valor en la posición i -ésima corresponde a un carácter numérico lo mandamos a la salida

Paso 3: Si este corresponde a un carácter que representa un operador o paréntesis, lo apilamos

Paso 4: Si encontramos un operador que es de menor importancia al último apilado desapilamos hasta dejar vacío y apilamos este nuevo valor

Paso 5: En caso de encontrar un paréntesis de cierre desapilamos todo lo anterior hasta encontrar el paréntesis de apertura, no se añaden estos caracteres (paréntesis) a nuestra salida

Pseudocódigo

Algorithm 1 Conversión Polaca

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

Ensure: $[537 * +21 + 9 * +]$

INICIO

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

Si $A[i]$ es número

Imprimir $A[i]$

Si no

Si $temp < A[i]$ o $A[i] =)$

 Desapilar

 push($A[i]$)

Si $A[i] != ($ o $A[i] =)$

FIN

Implementación con pila dinámica

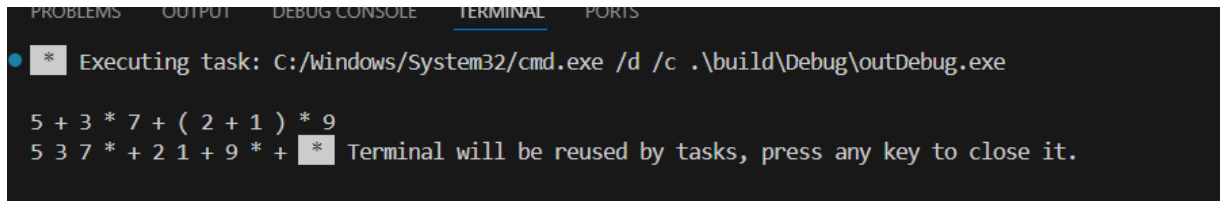
```
1 #include <stdio.h>
2 #include <malloc.h>
3 #include <stdbool.h>
4
5 struct nodo
6 {
7     char ele;
8     struct nodo *siguiente;
9 };
10
11 struct nodo *tope=NULL;
12
13 int isempty()
14 {
15     if(tope==NULL)
16     {
17         return 1;
18     }
19     return 0;
20 }
21
22 void imprimirarreglo(char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

```

53     tope=temp->siguiente;
54     char z=temp->ele;
55     free(temp);
56     return z;
57 }
58 }
59
60 void vaciar()
61 {
62     while(!isempty())
63     {
64         char z=pop();
65         if(z=='(')
66         {
67             break;
68         }
69     }
70 }
71
72 void polaca(char cad[13])
73 {
74     char temp='+';
75     for(int i=0;i<13;i++)
76     {
77         if((* (cad+i) <=57)&(* (cad+i) >=47))
78         {
79             printf("%c ",*(cad+i));
80         }
81         else
82         {
83             if((temp<*(cad+i))||(* (cad+i)=='('))
84             {
85                 vaciar();
86             }
87             push(* (cad+i));
88             if((* (cad+i) != '(')&(* (cad+i) != ')'))
89             {
90                 temp=*(cad+i);
91             }
92         }
93     }
94     vaciar();
95 }
96
97 int main()
98 {
99     char cad[13]={'5','+', '3','*', '7','+', '(', '2','+', '1',')','*', '9'};
100     imprimirarreglo(cad);
101     polaca(cad);
102 }

```

Compilación



```

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

5 + 3 * 7 + ( 2 + 1 ) * 9
5 3 7 * + 2 1 + 9 * + * 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 #define MAX 20
4
5 void imprimirarreglo(char c[13])
6 {
7     for (int l = 0; l < 13; l++)
8     {
9         printf("%c ", *(c+l));
10    }
11    printf("\n");
12 }
13
14 struct pila
15 {
16     char ele[MAX];
17     int tope;
18 };
19
20 void incializar(struct pila *p)
21 {
22     p->tope=-1;
23 }
24
25 int isempty(struct pila *p)
26 {
27     if(p->tope==-1)
28     {
29         return 1;
30     }
31     return 0;
32 }
33
34 bool isfull(struct pila *p)
35 {
36     if(p->tope==MAX-1)
37     {
38         return true;

```

```

39     }
40     return false;
41 }
42 }
43
44 void push(struct pila *p, char val)
45 {
46     if(isfull(p))
47     {
48         printf("La pila esta llena");
49     }
50     else
51     {
52         p->ele[++(p->tope)]=val;
53     }
54 }
55
56 char pop(struct pila *p)
57 {
58     if(isempty(p))
59     {
60         printf("La pila esta vacia");
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 z;
72     }
73 }
74 }
75
76 void vaciar(struct pila *p)
77 {
78     while(!isempty(p))
79     {
80         char z=pop(p);
81         if(z=='(')
82         {
83             break;
84         }
85     }
86 }
87
88 void polaca(char cad[13],struct pila *p)
89 {
90     char temp='+';
91     for(int i=0;i<13;i++)
92     {

```

```

93     if ((* (cad+i) <=57) & (* (cad+i) >=47))
94     {
95         printf("%c ", * (cad+i));
96     }
97     else
98     {
99         if ((temp < * (cad+i)) || (* (cad+i) == ')'))
100        {
101            vaciar(p);
102        }
103        push(p, * (cad+i));
104        if ((* (cad+i) != '(') & (* (cad+i) != ')'))
105        {
106            temp = * (cad+i);
107        }
108    }
109 }
110 vaciar(p);
111 }
112
113 int main()
114 {
115     struct pila p;
116     incializar(&p);
117     char cad[13] = {'5', '+', '3', '*', '7', '+', '(', '2', '+', '1', ')', '*', '9'};
118     imprimirarreglo(cad);
119     polaca(cad, &p);
120 }
121 }

```

Compliación

```

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

5 + 3 * 7 + ( 2 + 1 ) * 9
5 3 7 * + 2 1 + 9 * + * Terminal will be reused by tasks, press any key to close it.

```

Conclusión

A través de la implementación d euna estructura de datos logramos implementar el algoritmo de conversión Polaca, a su vez nos familiarizamos con el uso de las pilas tanto estáticas como dinámicas.