Pilhas
Stack

stack
/stak/

Prof. Bruno Travençolo
(com adaptações feitas por Paulo H. R. Gabriel)

1

Pilhas

- Estrutura de dados linear usada para armazenar e organizar dados
- Sequência de elementos do mesmo tipo
- Nessa estrutura somente temos conhecimento do último elemento inserido
  - Diferente de listas, que podemos percorrer todos os elementos.
  - Diferente de listas duplamente encadeadas, que podemos acessar o primeiro/último elemento via funções especializadas nessas tarefas
- A remoção de um elemento sempre será do elemento que estiver a menos tempo na pilha
  - Diferente de listas, que podemos remover elementos das extremidades por meio de funções especializadas
- É um tipo especial de lista, em que a inserção e a remoção são realizadas sempre na mesma extremidade
  - LIFO – Last In First Out (último a entrar, primeiro a sair)

2

Pilhas

- "Mas como faço então pra consultar um elemento 'no meio' da pilha?"
- Não faz. Se for necessário consultar algum elemento diferente do último inserido, isso significa que não é necessário usar uma pilha. Talvez uma lista serviria
- Vantagens: estrutura mais simples e mais flexível de implementar. Não há preocupação com vários ponteiros e também o número de operações é menor

3

Pilhas

- Operações de manipulação da pilha
  - Inserir um elemento na pilha
  - Remover um elemento da pilha
  - Acesso ao elemento da pilha
  - Verificar se a pilha está cheia ou vazia
- Operações relacionadas a estrutura da pilha
  - Criar a pilha
  - Destruir a pilha

4

Aplicação: parênteses e colchetes

- Bem formada?

```
( ( ) [ ( ) ] )
```

```
#define N 100
char pilha[N];
int t;

// Esta função devolve 1 se a string ASCII s
// contém uma sequência bem-formada de
// parênteses e colchetes e devolve 0 se
// a sequência é malformada.

int bemFormada (char s[])
{
   criapilha ();
   for (int i = 0; s[i] != '\0'; ++i) {
      char c;
      switch (s[i]) {
         case ')': if (pilhavazia ()) return 0;
                   c = desempilha ();
                   if (c != '(') return 0;
                   break;
         case ']': if (pilhavazia ()) return 0;
                   c = desempilha ();
                   if (c != '[') return 0;
                   break;
         default:  empilha (s[i]);
      }
   }
   return pilhavazia ();
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

5

Exemplo: Expressão matemática

- Motivação: avaliação de uma expressão matemática
  - 5 * (((9 + 8) * (4 * 6)) + 7)
- A pilha é a ED ideal para isso, pois permite guardar os valores intermediários das operações

```
push(5);
push(9);
push(8);
push(pop() + pop());
push(4);
push(6);
push(pop()*pop());
push(pop()*pop());
push(7);
push(pop()+pop());
push(pop()*pop());
```

Fonte: Algorithms in C Sedgewick

6

Exemplo: Expressão matemática

- Essa ordem de cálculo exige que os operandos (os números) estejam na pilha antes do operadores (+ ou * neste exemplo)
- 9 e 8 estão na pilha para só depois ser feito a operação de soma

```
push(5);
push(9);
push(8);
push(pop() + pop());
push(4);
push(6);
push(pop()*pop());
push(pop()*pop());
push(7);
push(pop()+pop());
push(pop()*pop());
printf("%d\n", pop);
```

Fonte: Algorithms in C Sedgewick

Exemplo: Expressão matemática

- Qualquer operação aritmética pode ser reescrita
  - (5 * (((9 + 8) * (4 * 6)) + 7))
- Passa a ser
  - 5 9 8 + 4 6 * * 7 + *
- Essa notação é chamada de *Reverse Polish* ou **postfix (posfixa)**. A primeira notação, que estamos acostumados, é chamada de **infix (infixa)**
- Na notação **postfix** não utilizamos parênteses

```
push(5);
push(9);
push(8);
push(pop() + pop());
push(4);
push(6);
push(pop()*pop());
push(pop()*pop());
push(7);
push(pop()+pop());
push(pop()*pop());
printf("%d\n", pop());
```

Fonte: Algorithms in C Sedgewick

Exemplo: Expressão matemática

- Converter infixa para posfixa

infixa	posfixa
(A+B*C)	ABC*+
(A*(B+C)/D-E)	ABC+*D/E-
(A+B*(C-D*(E-F)-G*H)-I*3)	ABCDEF-*-GH*-*+I3*-
(A+B*C/D*E-F)	ABC*D/E*+F-
(A+B+C*D-E*F*G)	AB+CD*+EF*G*-
(A+(B-(C+(D-(E+F)))))	ABCDEF+-+-+
(A*(B+(C*(D+(E*(F+G))))))	ABCDEFG+*+*+*

Note que os operandos (A, B, C, etc.) aparecem na mesma ordem na expressão infixa e na correspondente expressão posfixa. Note também que a notação posfixa dispensa parênteses e regras de precedência entre operadores (como a precedência de * sobre + por exemplo), que são indispensáveis na notação infixa.

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

Exemplo: Expressão matemática

- Todo parêntese esquerdo é colocado na pilha.
- Ao encontrar um parêntese direito, o algoritmo desempilha tudo até encontrar um parêntese esquerdo, que também é desempilhado.
- Ao encontrar um + ou um - , o algoritmo desempilha tudo até encontrar um parêntese esquerdo, que não é desempilhado.
- Ao encontrar um * ou um / , o algoritmo desempilha tudo até encontrar um parêntese esquerdo ou um + ou um - .
- Constantes e variáveis são transferidos diretamente de inf para a expressão posfixa. Os operadores são transferidos quando desempilhados nos passos descritos acima

infix | (| A | * | (| B | * | C | + | D |) |) |

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

Exemplo: Expressão matemática

- Traduzir para notação posfixa a expressão infixa armazenada em uma string
- (A*(B*C+D))
- Supor que a expressão esteja em uma string em C

infix | (| A | * | (| B | * | C | + | D |) |) |

- Devemos criar um vetor para armazenar o resultados posfixo

posfix | | | | | | | | | | | |

- E uma pilha pra avaliar expressão

stack | | | | | | | | | | |

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

Exemplo: Expressão matemática

inf | (| A | * | (| B | * | C | + | D |) |) |

posf | | | | | | | | | | | |

stack | | | | | | | | | | |

```
char *infixaParaPosfixa (char *inf) {
   int n = strlen (inf);
   char *posf;
   posf = malloc ((n+1) * sizeof (char));
   criapilha ();
```

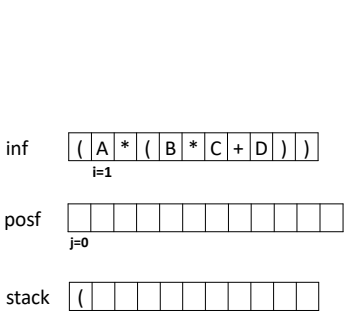
inf | (| A | * | (| B | * | C | + | D |) |) |
*

posf | | | | | | | | | | | |

```
   empilha (inf[0]); // empilha '('
```

stack | (| | | | | | | | | |

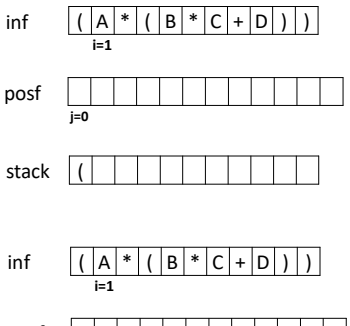
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

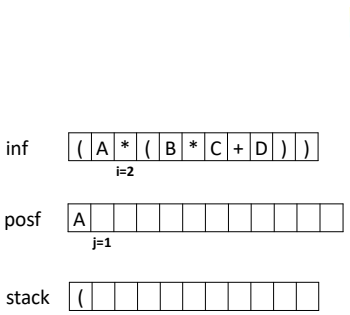
13




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

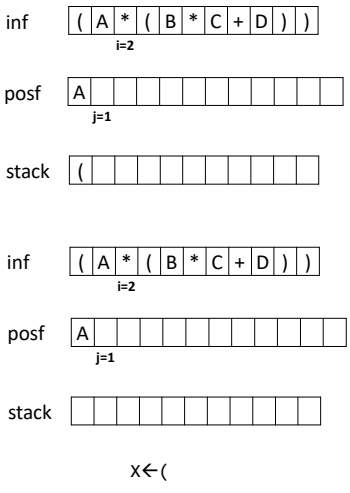
14




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

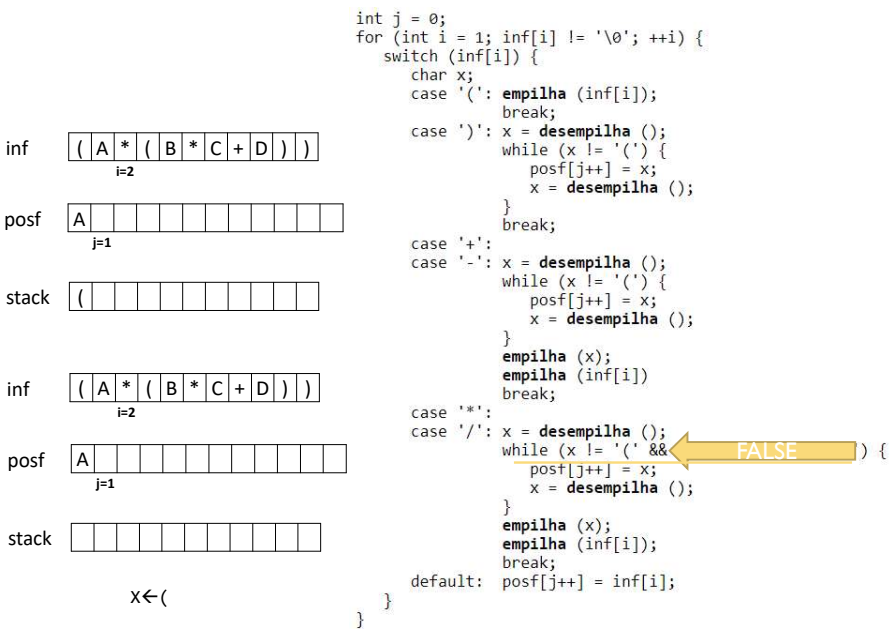
15




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

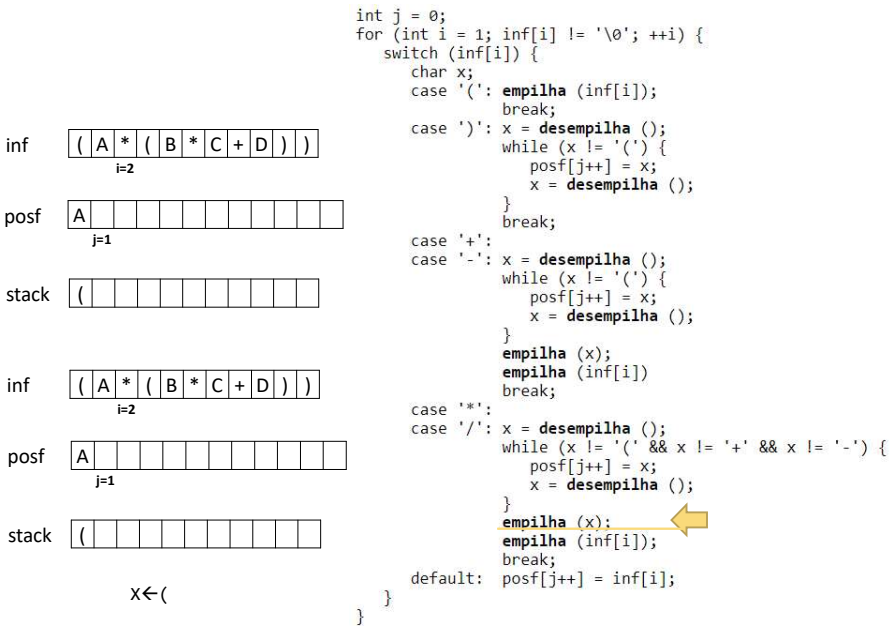
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

16



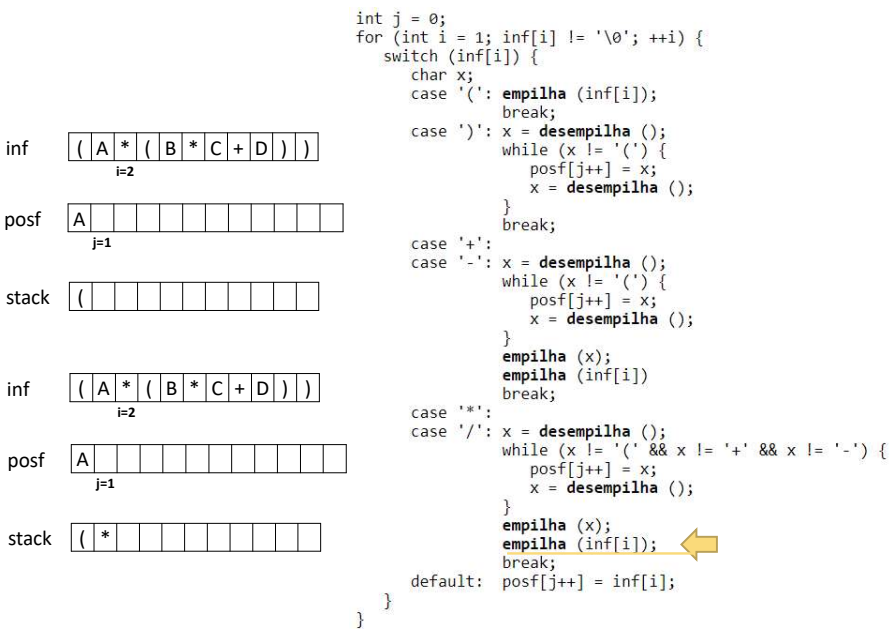
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

17



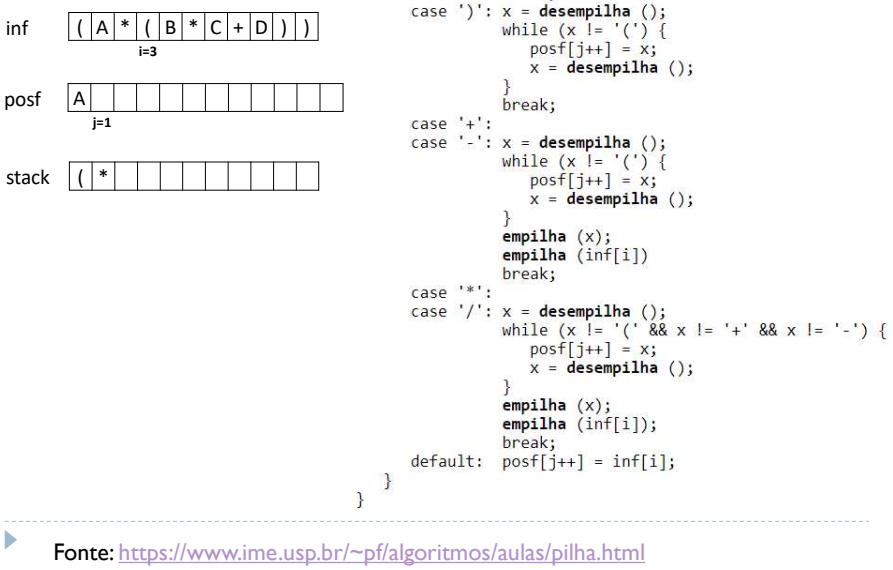
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

18



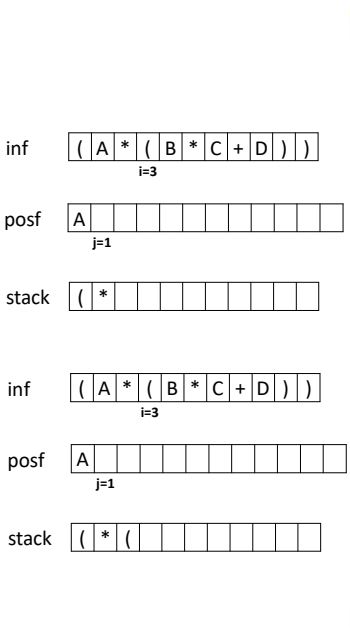

```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

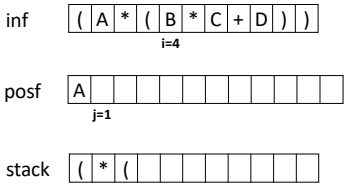
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

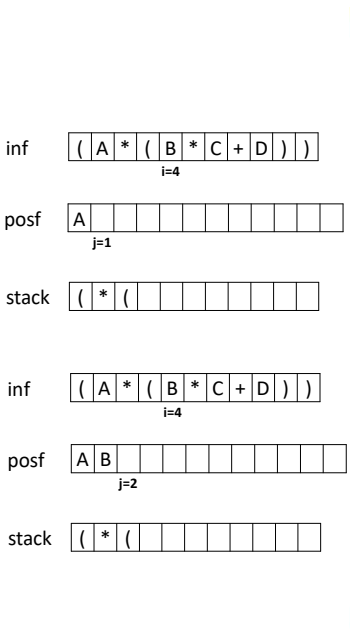
21




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

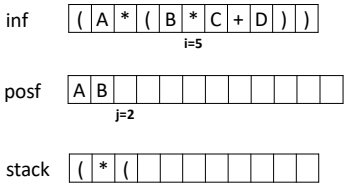
22




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

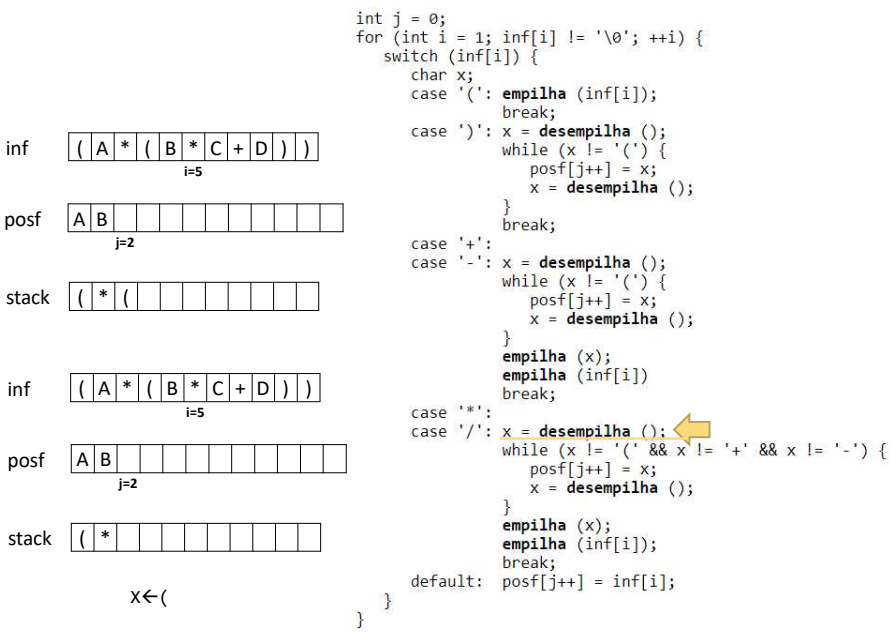
23




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

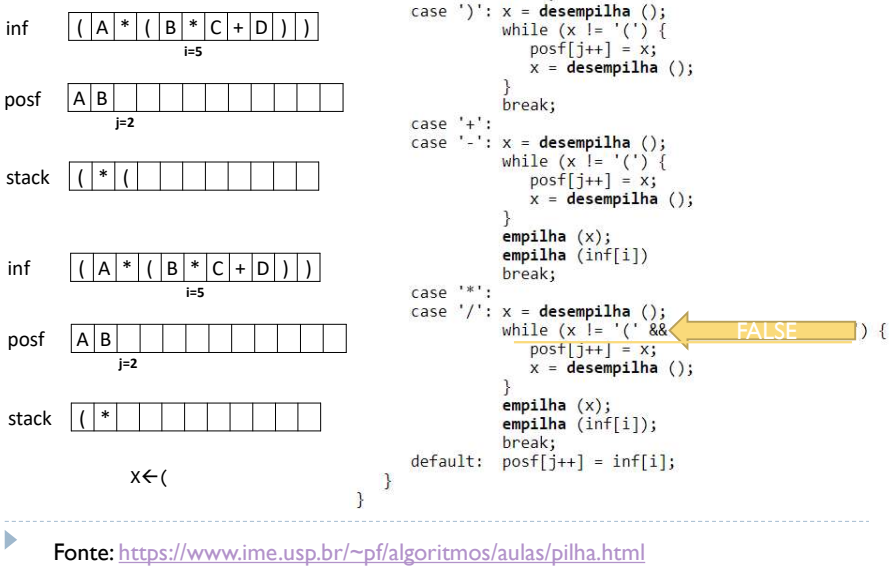
24




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

25

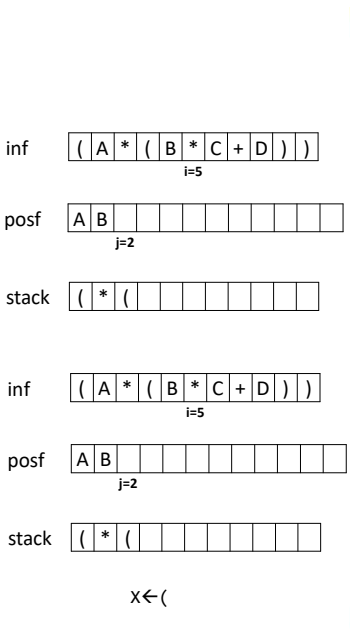



```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' &&          ) {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

FALSE

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

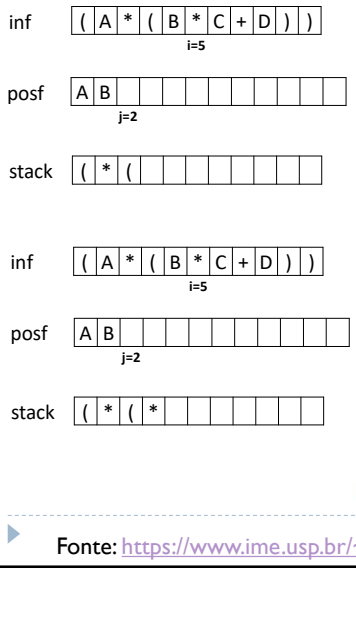
26




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

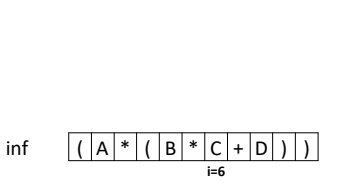
27




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

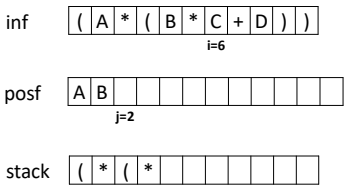
28




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

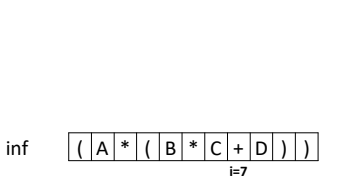
29




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

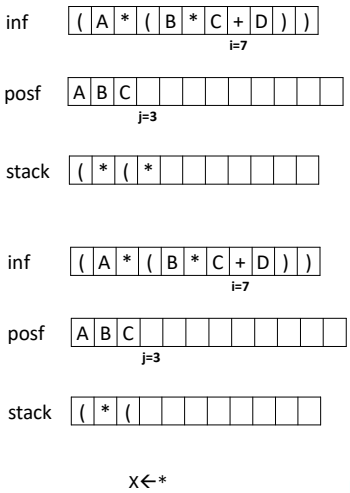
30




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

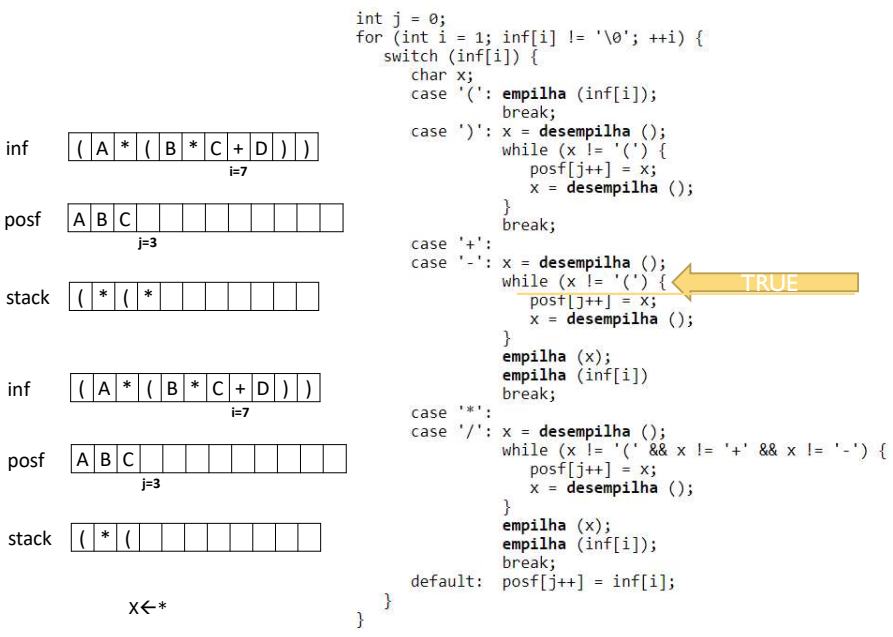
31




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

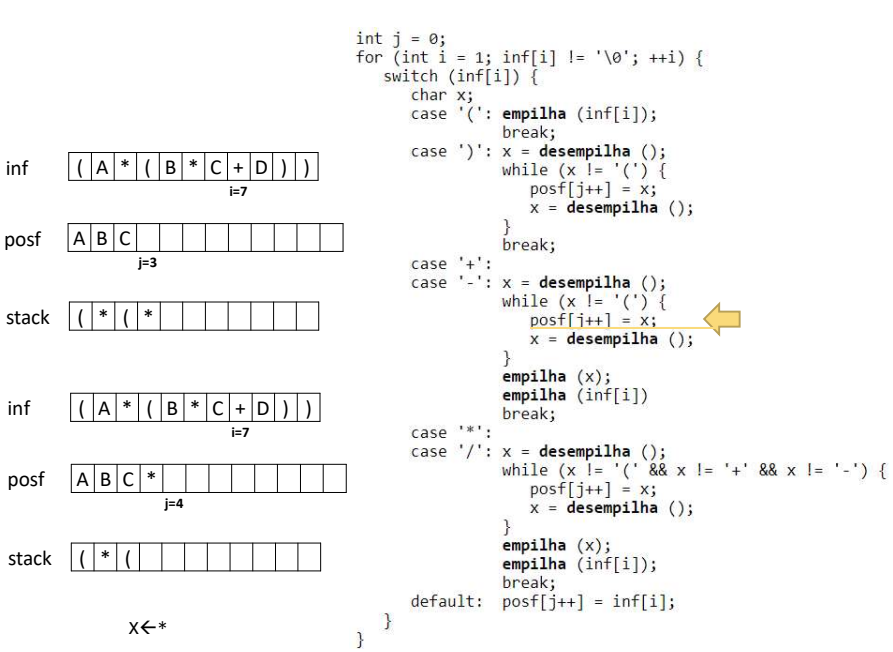
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

32



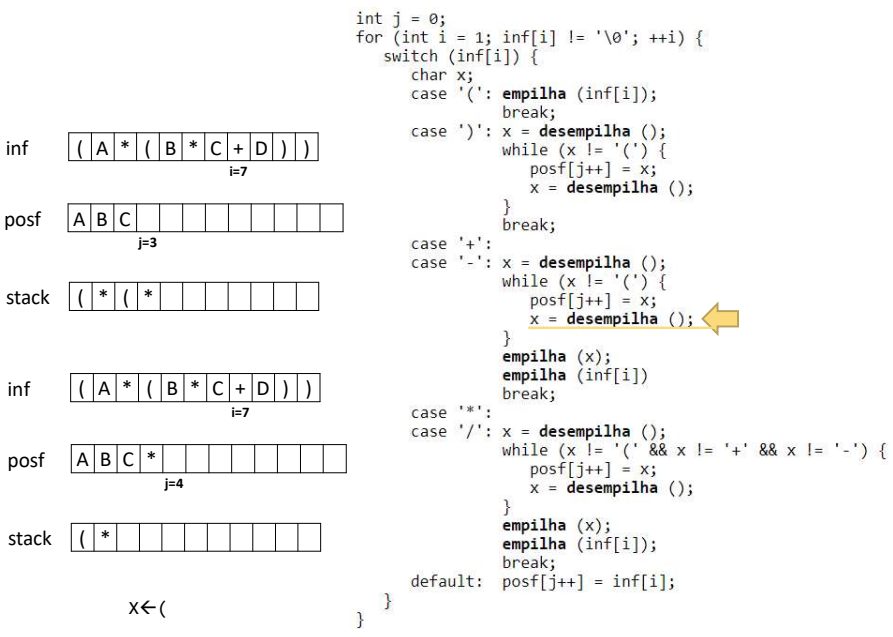
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

33



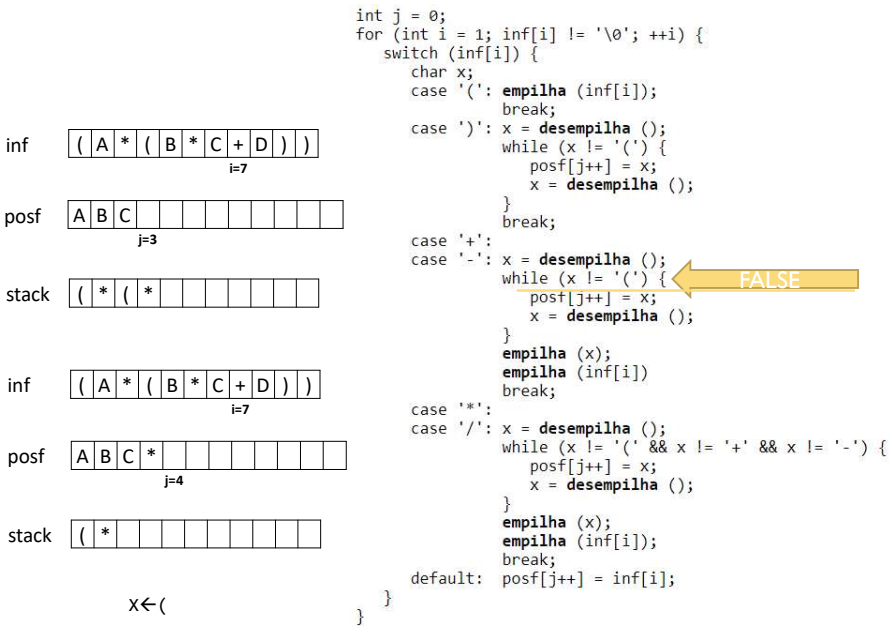
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

34




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

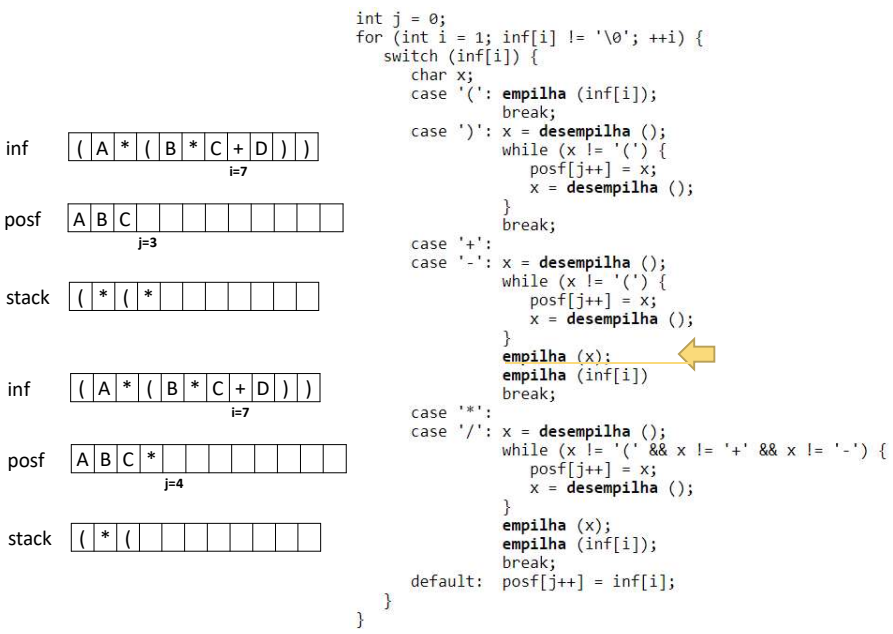
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

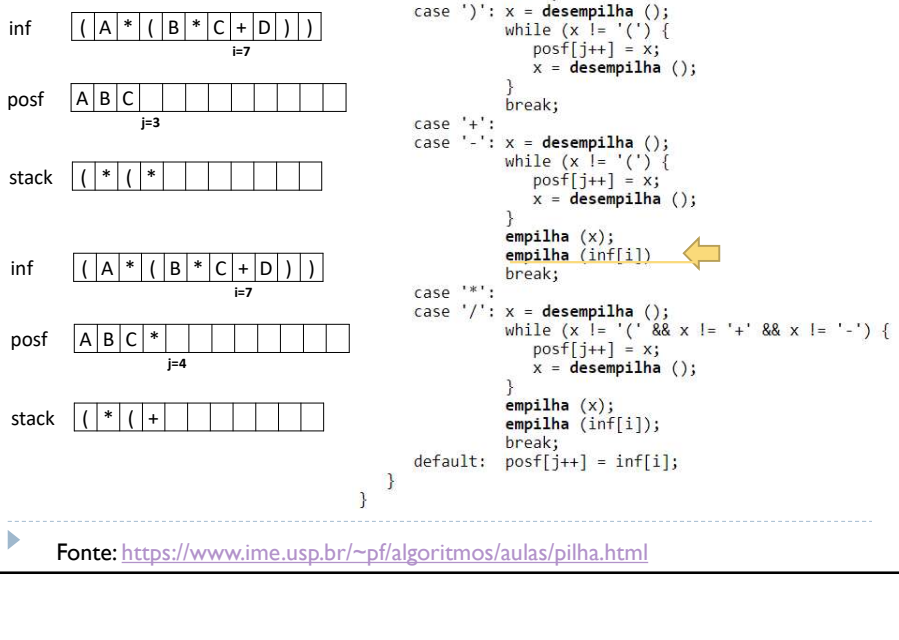
FALSE

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html



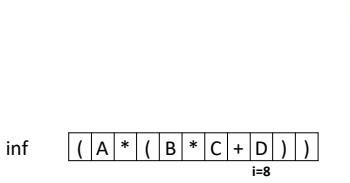

```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html



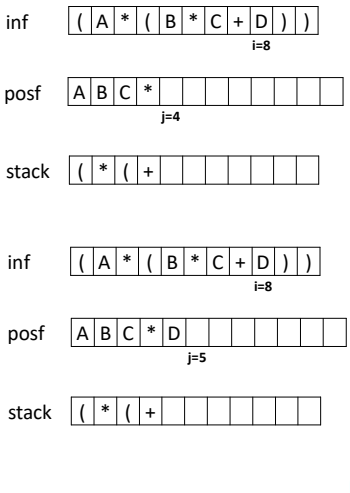

```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html



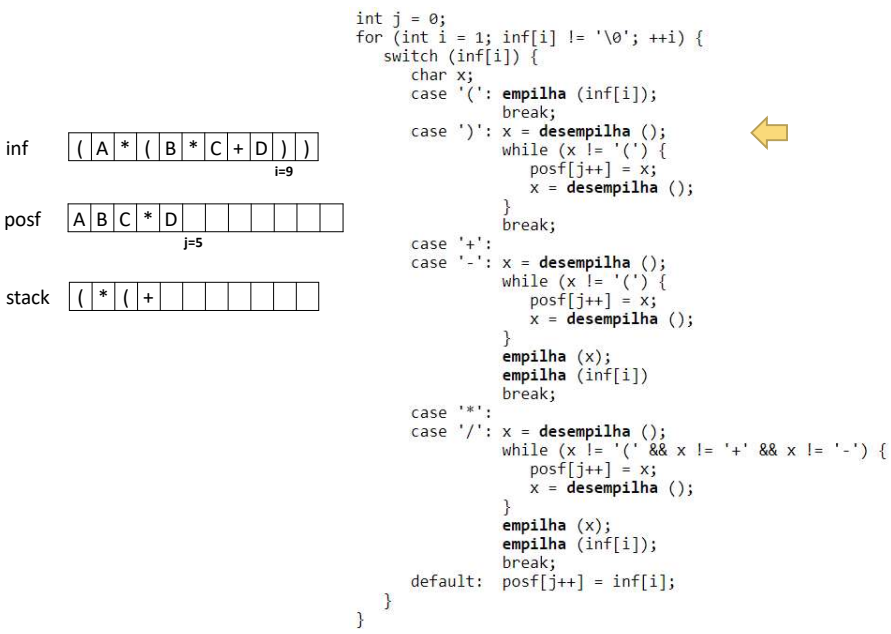

```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

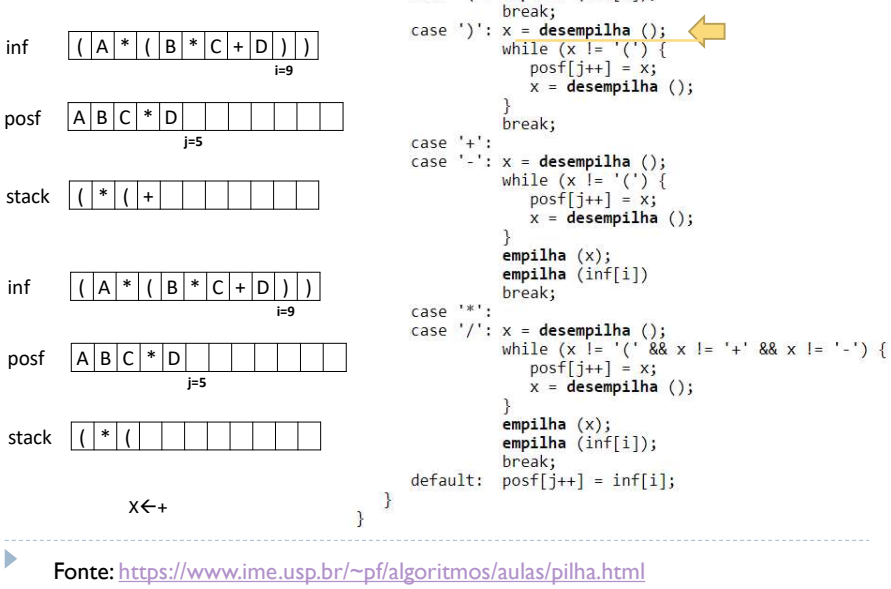
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

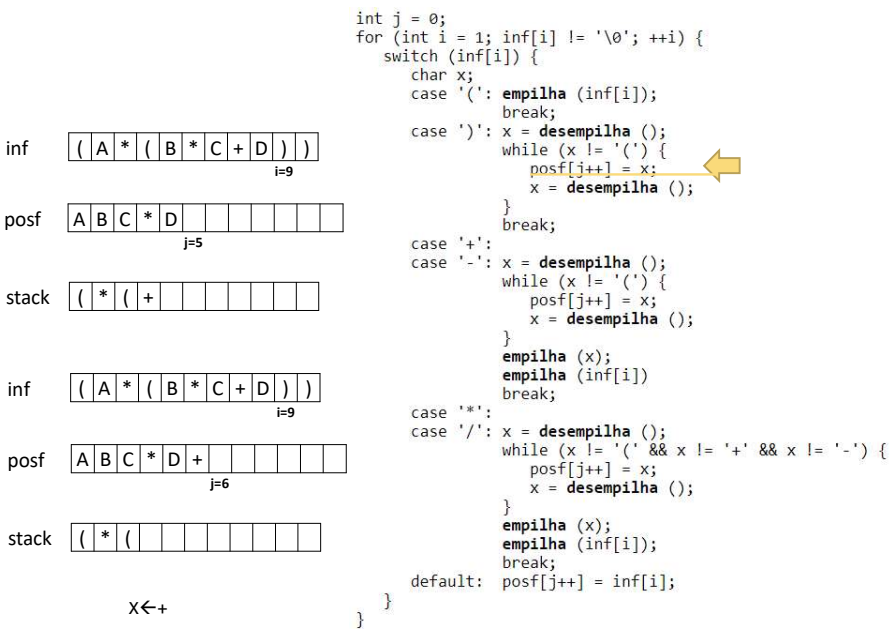
41




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

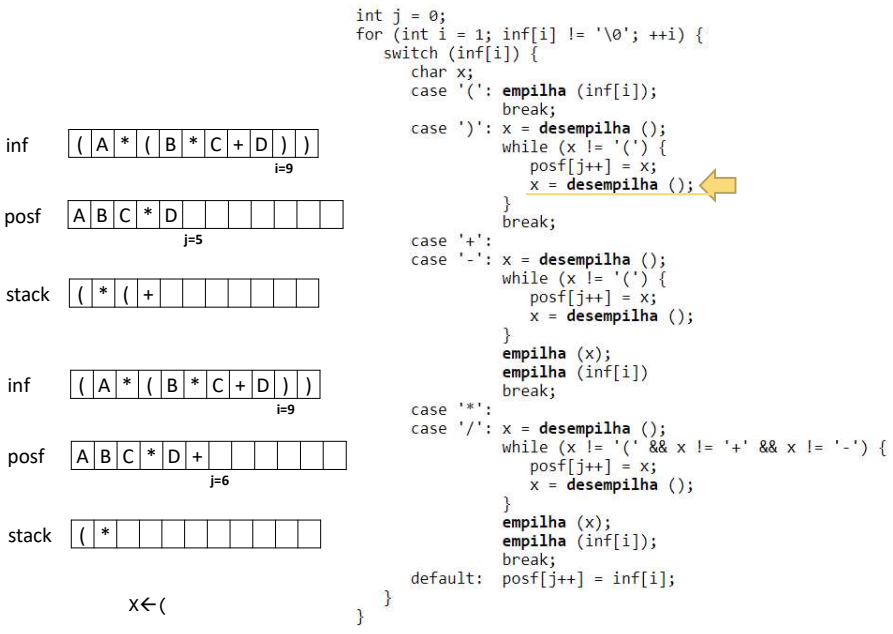
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

42



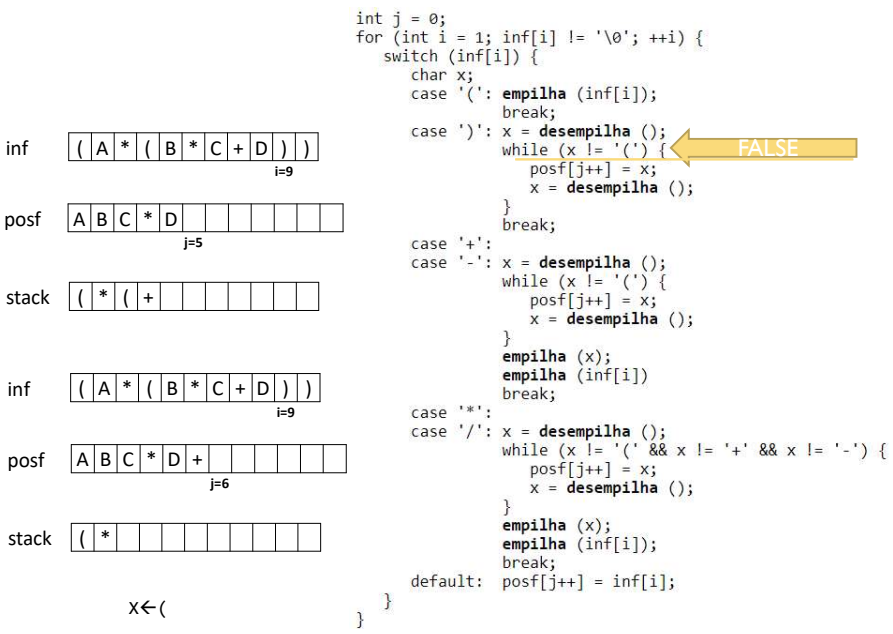
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

43



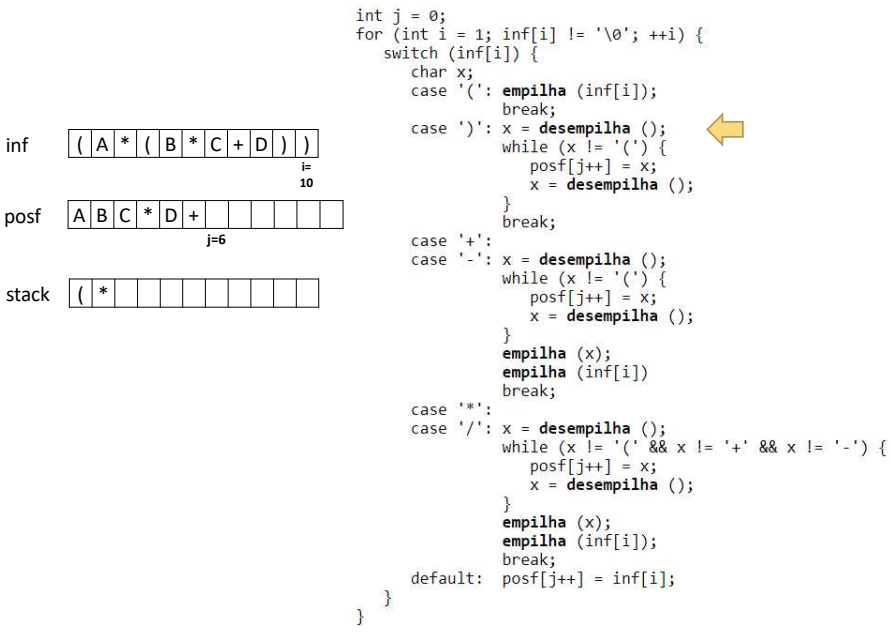
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

44



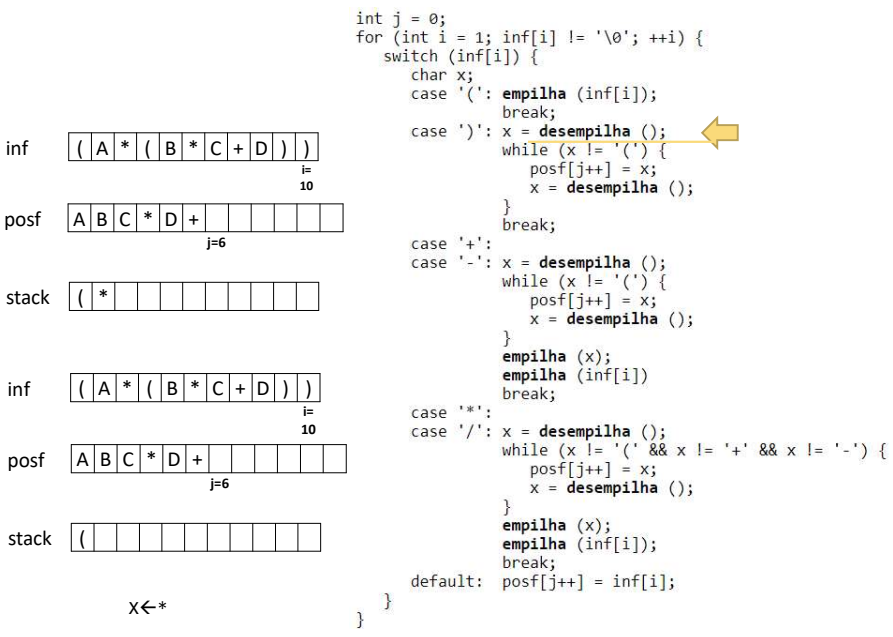
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

45



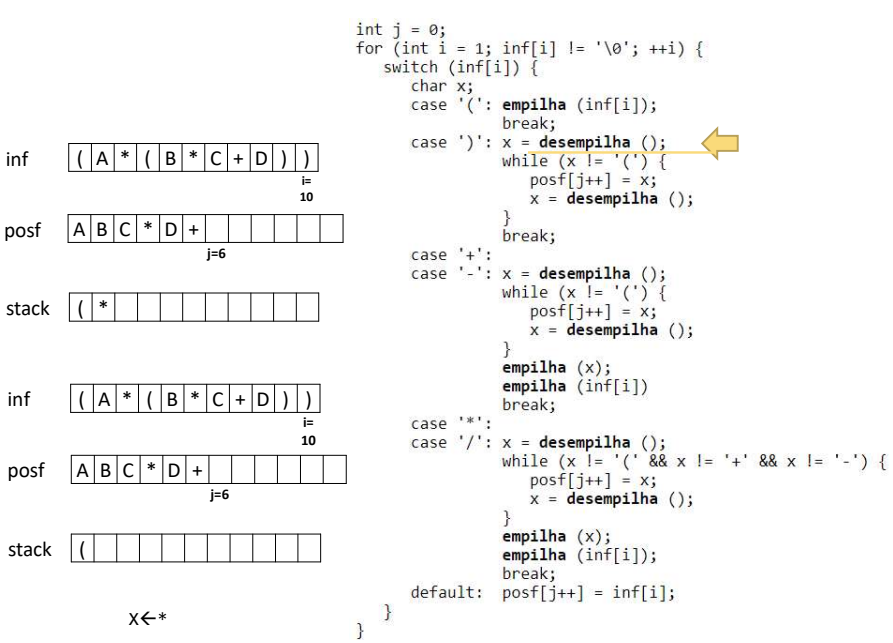
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

46



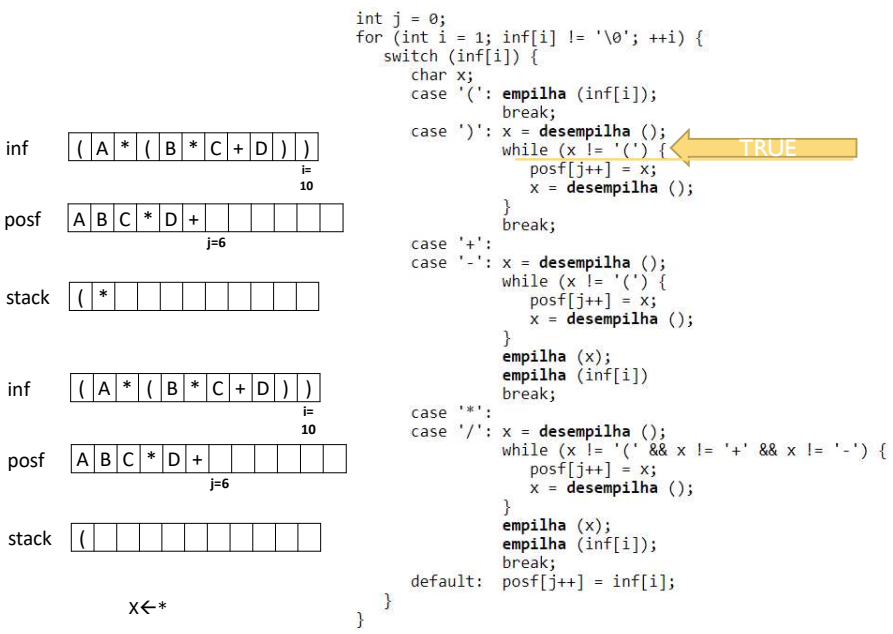
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

47



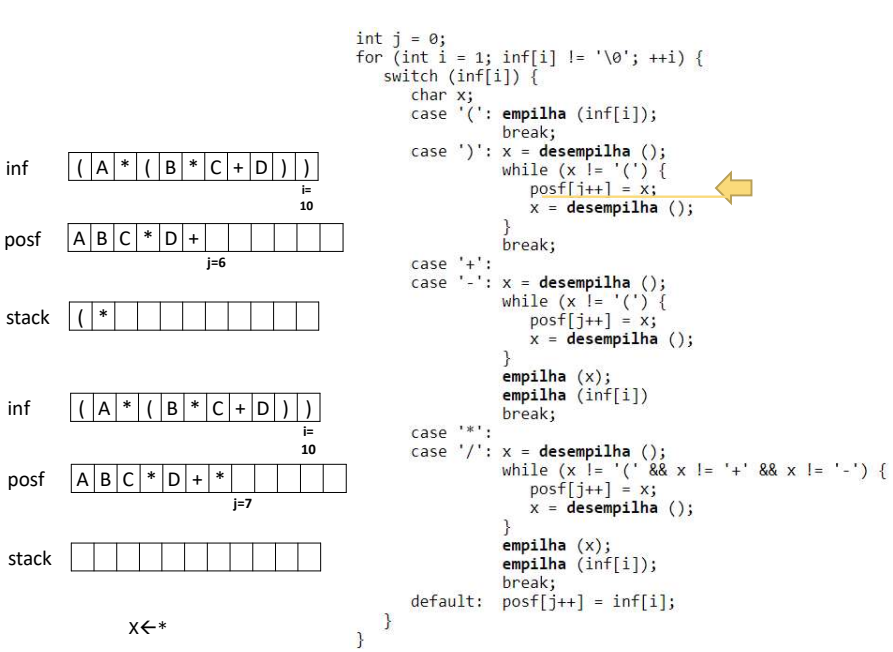
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

48



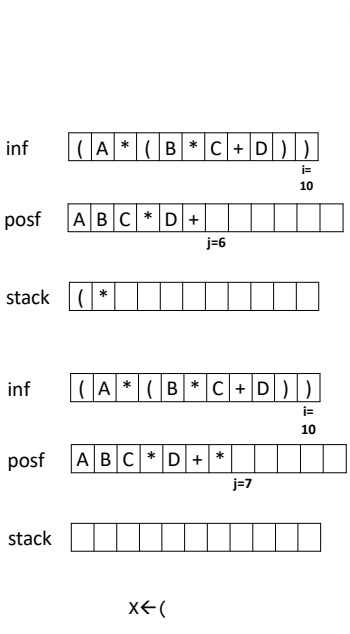
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

49



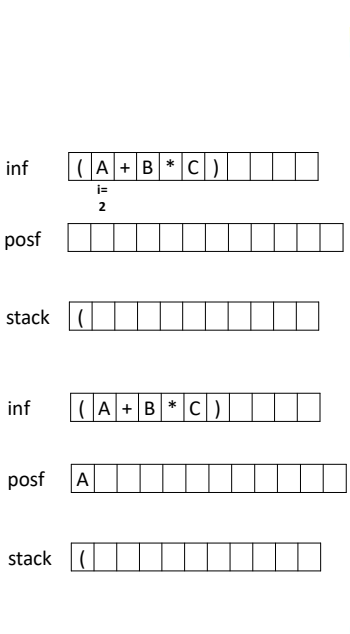
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

50



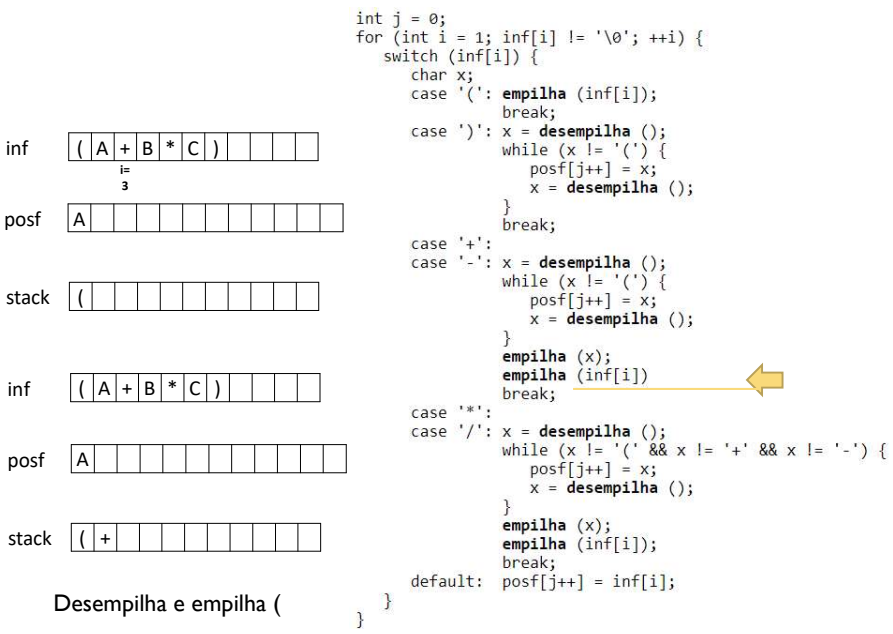

```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html



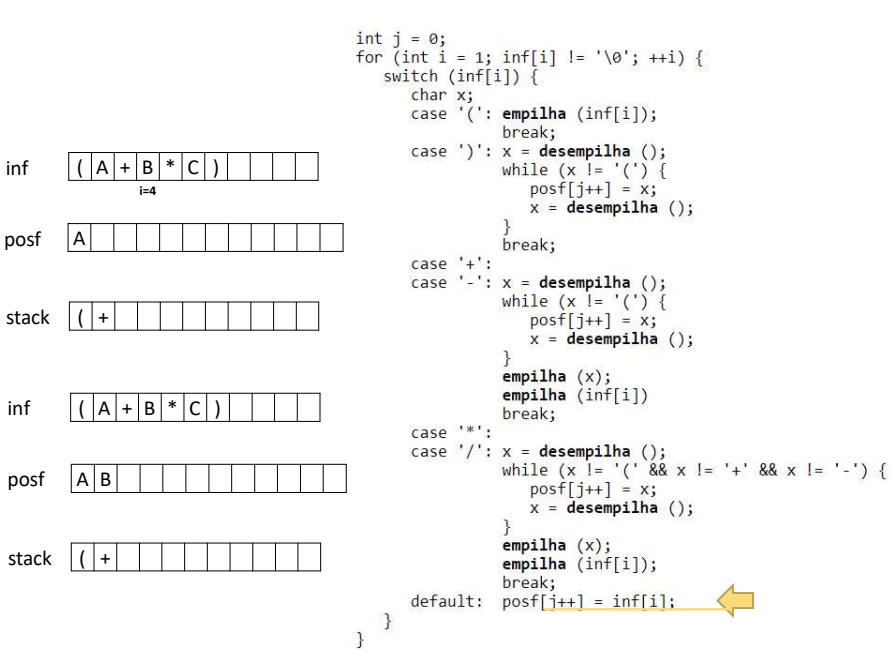

```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html



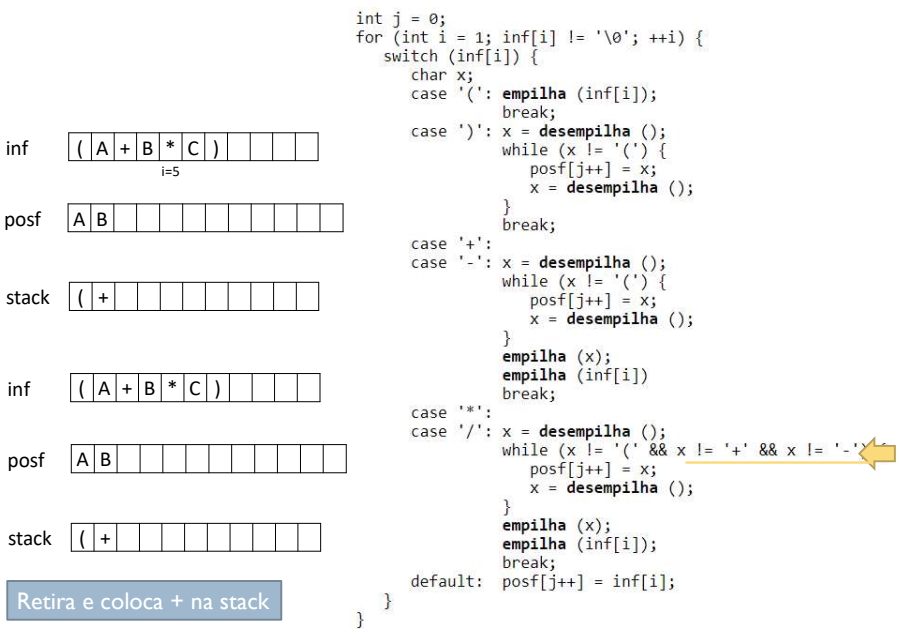

```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html



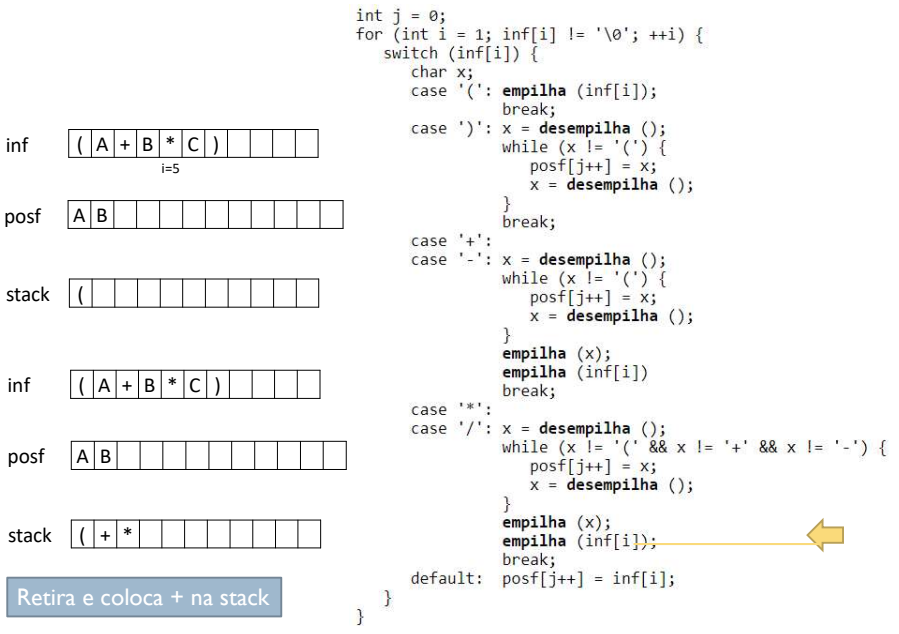

```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html



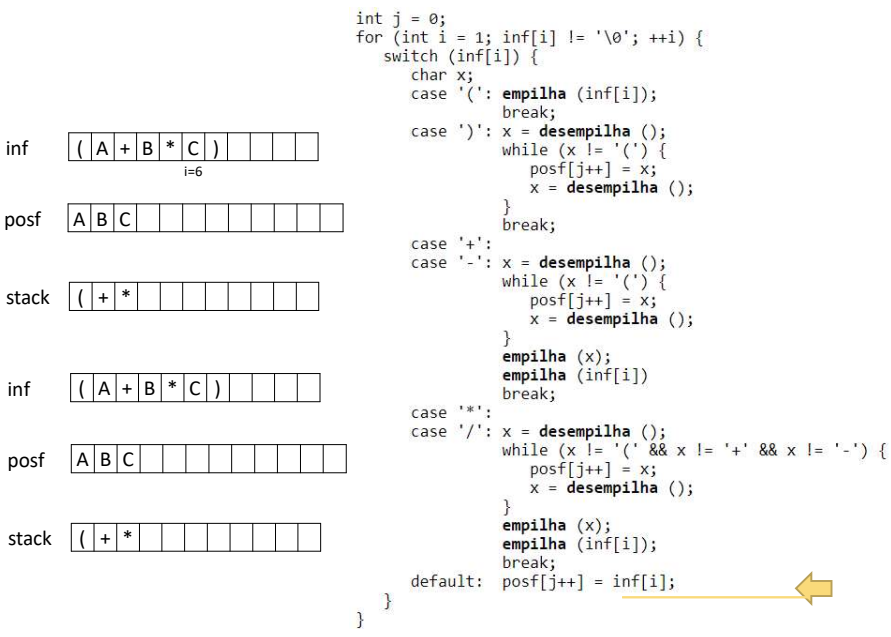
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

55



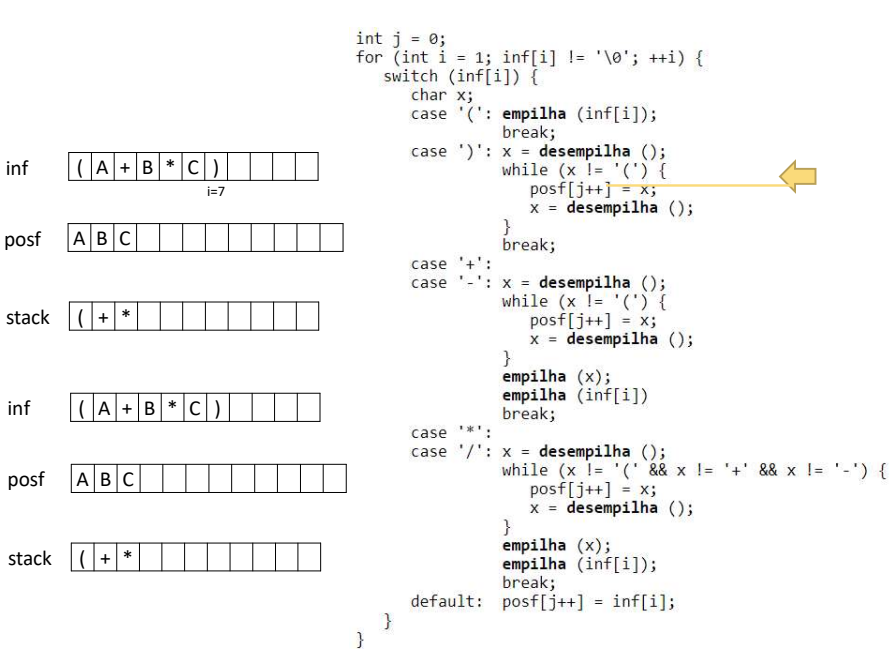
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

56



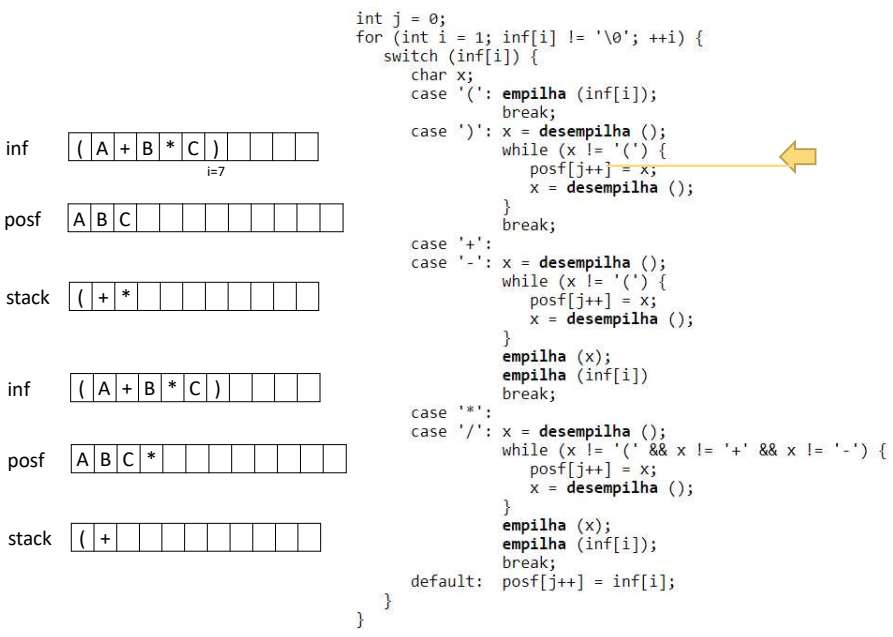
Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

57



Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

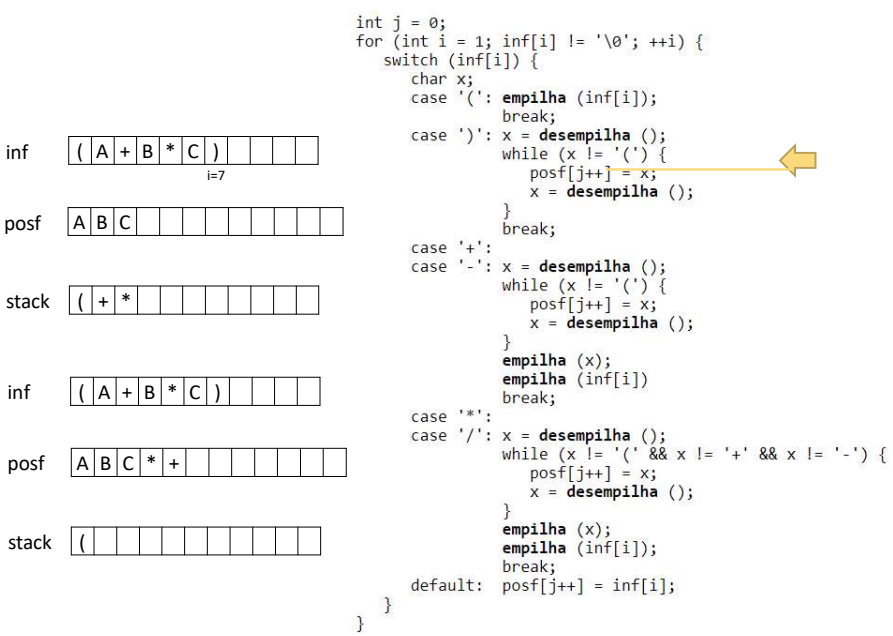
58




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

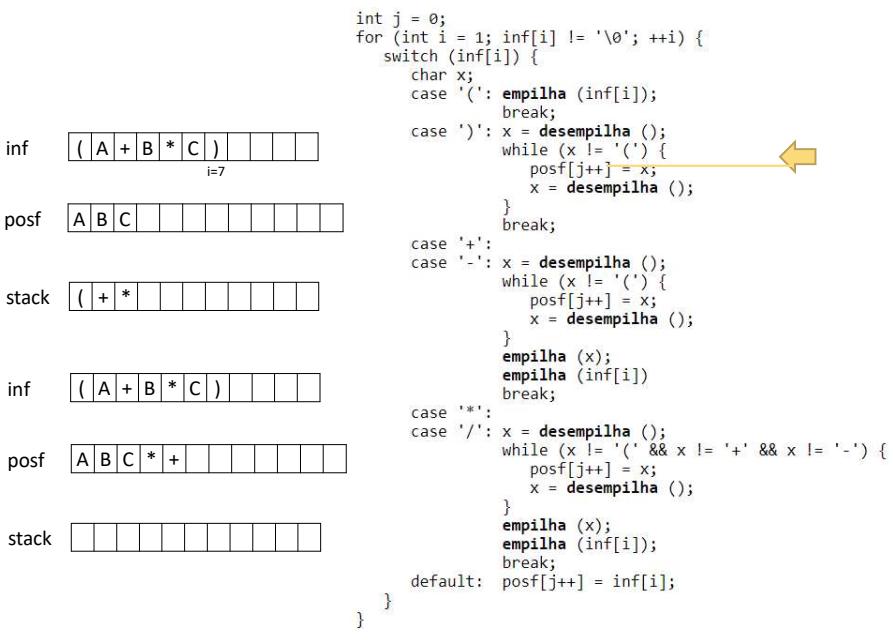
59




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

60




```
int j = 0;
for (int i = 1; inf[i] != '\0'; ++i) {
   switch (inf[i]) {
      char x;
      case '(': empilha (inf[i]);
                break;
      case ')': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                break;
      case '+':
      case '-': x = desempilha ();
                while (x != '(') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i])
                break;
      case '*':
      case '/': x = desempilha ();
                while (x != '(' && x != '+' && x != '-') {
                   posf[j++] = x;
                   x = desempilha ();
                }
                empilha (x);
                empilha (inf[i]);
                break;
      default:  posf[j++] = inf[i];
   }
}
```

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pilha.html

61

Outro exemplo: chamada de sub-rotinas

Rotina A	Rotina B	Rotina C	Rotina D
1 print "A"	1 call C	1 print "C"	1 print "D"
2 call C	2 print "B"	2 call D	2 return
3 call B	3 call D	3 return	
4 call D	4 call C		
5 return	5 return		

Qual o resultado da execução da rotina A?

Fonte: Adaptado das notas de aula do prof. Thiago A. S. Pardo (ICMC/USP)

62

Outro exemplo: chamada de sub-rotinas

1. Um computador está executando a rotina X e, durante a execução de X, encontra uma chamada à rotina Y
2. Interrompe a execução de X e se inicia a execução de Y
3. Quando termina a execução de Y, o computador deve saber o que fazer, isto é, onde voltar na rotina X

Fonte: Adaptado das notas de aula do prof. Thiago A. S. Pardo (ICMC/USP)

63

Outro exemplo: chamada de sub-rotinas

- Dificuldade
  - O que estava sendo executado quando uma sub-rotina foi interrompida?
  - Para onde voltar agora que se chegou ao fim de uma sub-rotina?
- Solução
  - A cada chamada de sub-rotina, armazenar o endereço de retorno (rotina e número da linha, por exemplo)
  - Como armazenar o endereço de retorno de chamadas sucessivas: pilha

Fonte: Adaptado das notas de aula do prof. Thiago A. S. Pardo (ICMC/USP)

64

Outro exemplo: chamada de sub-rotinas

- Dificuldade
  - O que estava sendo executado quando uma sub-rotina foi interrompida?
  - Para onde voltar agora que se chegou ao fim de uma sub-rotina?
- Solução
  - A cada chamada de sub-rotina, armazenar o endereço de retorno (rotina e número da linha, por exemplo)
  - Como armazenar o endereço de retorno de chamadas sucessivas: pilha

Fonte: Adaptado das notas de aula do prof. Thiago A. S. Pardo (ICMC/USP)

Outro exemplo: chamada de sub-rotinas

- A cada comando **call**
  - Empilha (*push*) o endereço para retornar depois
  - Passa a executar a nova sub-rotina
- A cada comando **return**
  - Desempilha (*pop*) o último endereço armazenado
  - Passa a executar a partir do endereço desempilhado

Fonte: Adaptado das notas de aula do prof. Thiago A. S. Pardo (ICMC/USP)