Ficha 1.17

1. Implemente um ADT Stack de acordo com os seguintes protótipos:

- ```
  /*
  DESCRIÇÃO: Inicializa uma stack com a capacidade n
  PRÉ: n > 0;
  PÓS: (STACKempty() == 1) && (STACKfull()==0)
  EXCEÇÃO: aborta caso não haja memória

    Primeira versão:      2012.06.03 por António dos Anjos
                          email: antoniodosanjos@gmail.com
    Alterações:
  */
  void STACKinit(int n);
  ```

- ```
  /*
  DESCRIÇÃO: Devolve 1 se a stack estiver vazia, ou zero caso contrário
    Primeira versão:      2012.06.03 por António dos Anjos
                          email: antoniodosanjos@gmail.com
    Alterações:
  */
  int STACKempty(void);
  ```

- ```
  /*
  DESCRIÇÃO: Devolve 1 se a stack estiver cheia, ou zero caso contrário
    Primeira versão:      2012.06.03 por António dos Anjos
                          email: antoniodosanjos@gmail.com
    Alterações:
  */
  int STACKfull(void);
  ```

- ```
  /*
  DESCRIÇÃO: Retorna e retira o último elemento inserido na stack
  PRÉ: STACKempty() == 0
  PÓS: STACKfull()== 0

    Primeira versão:      2012.06.03 por António dos Anjos
                          email: antoniodosanjos@gmail.com
    Alterações:
  */
  s_type_t STACKpop(void);
  ```

- ```
  /*
  DESCRIÇÃO: Retorna o último elemento inserido na stack, sem modificar o
  estado da stack
  PRÉ: STACKempty() == 0
  PÓS: STACK == old STACK (onde STACK representa o estado da stack)

    Primeira versão:      2012.06.03 por António dos Anjos
                          email: antoniodosanjos@gmail.com
  Alterações:
  */
  s_type_t STACKtop(void);
  ```
  ```

- ```
  /*
  DESCRIÇÃO: Insere um elemento do tipo s_type_t na stack
  PRÉ: STACKfull() == 0
  PÓS: (STACKempty()== 0) && (item == STACKtop())

    Primeira versão:      2012.06.03 por António dos Anjos
                          email: antoniodosanjos@gmail.com
    Alterações:
  */
  void STACKpush(s_type_t item);
  ```

- ```
  /*
  DESCRIÇÃO: Liberta o espaço alocado por STACKinit()
  PRÉ: STACKinit() foi chamada && STACKkill() não foi chamada
  PÓS: O espaço alocado por STACKinit fica livre

    Primeira versão:      2012.06.03 por António dos Anjos
                          email: antoniodosanjos@gmail.com
    Alterações:
  */
  void STACKkill(void);
  ```

2. Utilizando a ADT Stack, implemente em ANSI C o algoritmo apresentado abaixo (encontra-se em pseudocódigo muito simplificado) para converter expressões aritméticas infixas em sufixas. Este algoritmo assume que:
   - A expressão de input (a partir da linha de comandos) apenas tem números de um dígito;
   - Além dos dígitos, apenas se admitem os operadores +, -, *, / e %.

   Exemplo de utilização: in2suf 2+5*3 → *2 5 3 * +*

```
Algoritmo:

Percorrer a expressão da esquerda para a direita;
      Se for dígito, mostrar;
      Se for operador:
            Se a Stack estiver vazia, STACKpush(operador);
            Senão,
                  Se (preced(operador) > preced(STACKtop())) e !STACKfull()
                        Então STACKpush(op);
                  Senão,
                        Enquanto (preced(operador) <= preced(STACKtop())) e !STACKempty()
                              Mostrar STACKpop();
                        STACKpush(operador);
      Mostrar o resto da Stack
```

onde, *preced(operador)* é uma função que devolve a precedência do operador em causa.
NOTA: não é necessário converter os dígitos para inteiros, já que não vamos fazer cálculos.