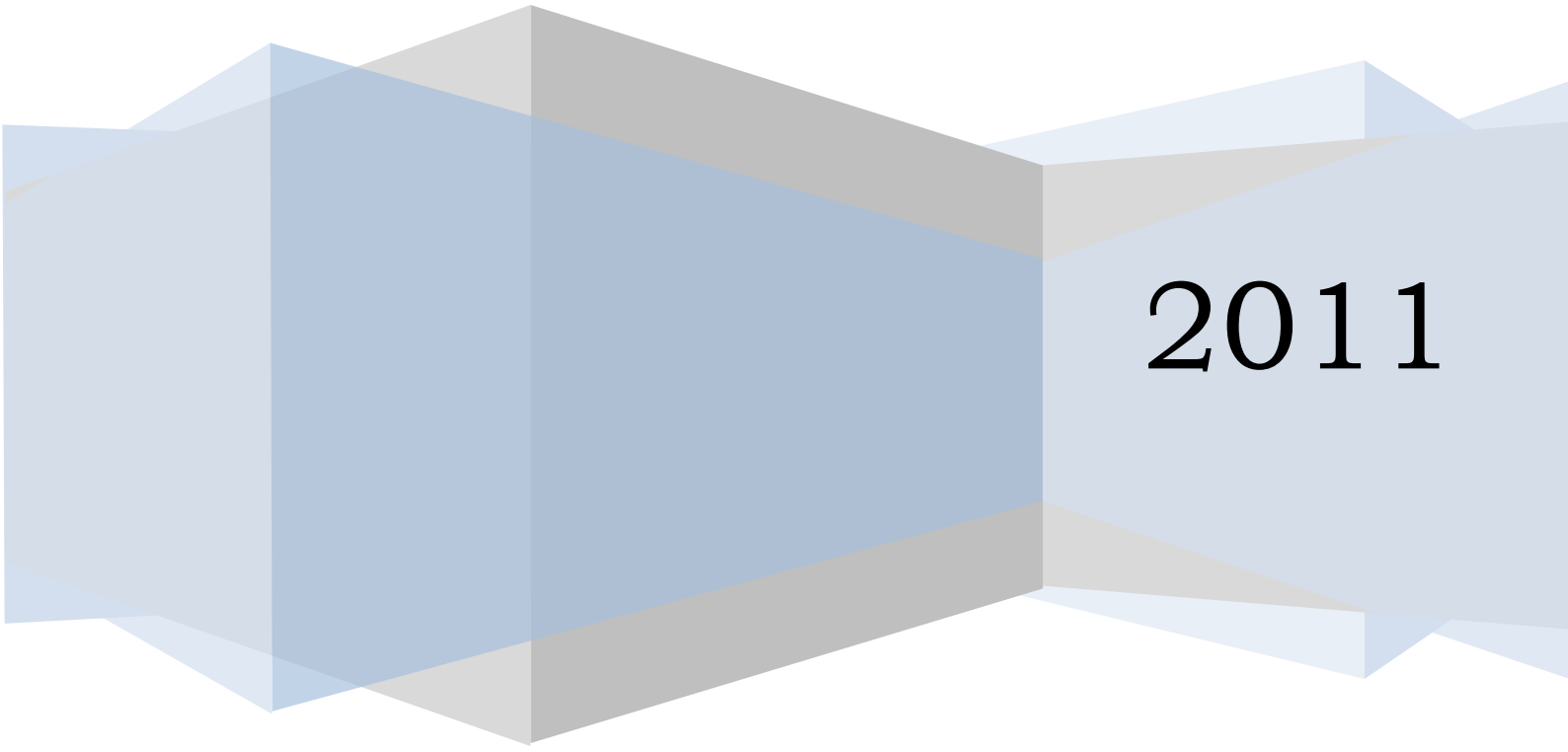
Flávio Augusto de Freitas
Introdução à Programação em Linguagem C/C++

http://flavioaf.blogspot.com

C/C++

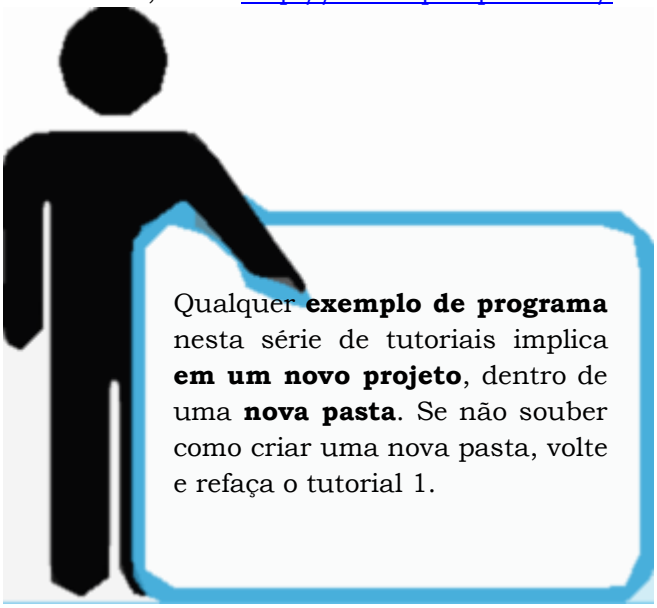
Tutorial 8 (usando Dev-C++ versão 4.9.9.2)



2011

1 INTRODUÇÃO

Esta série de tutoriais foi escrita usando o **Microsoft Windows 7 Ultimate** e o **Bloodshed Dev-C++** versão 4.9.9.2, que pode ser baixada em http://www.bloodshed.net. Se alguém quiser adquirir mais conhecimentos e quiser aprofundar no assunto, visite http://www.cplusplus.com/.



Qualquer **exemplo de programa** nesta série de tutoriais implica **em um novo projeto**, dentro de uma **nova pasta**. Se não souber como criar uma nova pasta, volte e refaça o tutorial 1.

2 O QUE É UMA PILHA?

Uma pilha é uma estrutura de dados que implementa a filosofia LIFO (***Last In, First Out)***, ou seja, o último item a ser inserido na pilha é o primeiro a poder ser retirado, pois é o que se encontra por "cima".

Uma analogia que pode ser feita consiste em pensar em termos de uma pilha de livros.

Se apenas conseguir manipular um livro de cada vez, como é que faço para obter o livro de baixo?

Sim, a resposta é ir desempilhando os livros de cima, um a um, até o livro pretendido ficar acessível.

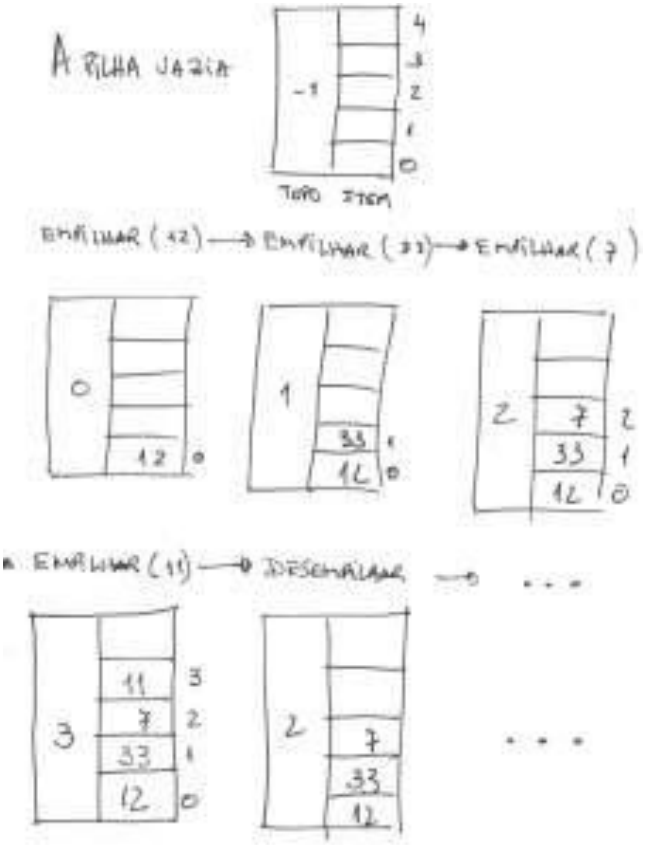
São muitas as formas de implementação de pilhas e nas mais variadas linguagens de programação.

A que aqui vou abordar implementa uma pilha sobre uma estrutura com dois membros, um dois quais é um *array* (vetor) que irá armazenar a pilha propriamente dita.

topo: É um inteiro que me indica a posição do topo da pilha. O valor -1 significa que a pilha está vazia.

item: É um *array* de inteiros de tamanho determinado por **#define tamanho 5**.

A imagem seguinte tenta ilustrar as operações que foram executadas na função main.



Por questões de simplificação as funções empilhar e desempilhar não efetuam por si só o teste às situações "pilha vazia" e "pilha cheia", pelo que deverão ser invocadas em conjunto com as funções aqui disponibilizadas para esse efeito.

O que quero eu dizer?

- Só se pode **empilhar** numa pilha que **não está cheia**.
- Só se pode **desempilhar** numa pilha que **não está vazia**.

3 PROGRAMA-EXEMPLO

```
#include <iostream>
#define tamanho 5

using namespace std;

typedef struct{
      int topo ;
      int item [tamanho] ;
}PILHA;

void iniciaPilha (PILHA &p) {
     p.topo = -1 ;
}

bool pilhaVazia(PILHA p){
  if(p.topo == -1 )
    return true;
  else
    return false;
}

bool pilhaCheia(PILHA p){
  if (p.topo == tamanho-1)
    return true;
  else
    return false;
}

void empilha(PILHA &p, int x){
  p.item[++p.topo] = x;
}

int desempilha(PILHA &p){
  return (p.item[p.topo--]);
}

//O mais importante já passou.
//Este código agora é só para testar.
int main(){
  PILHA s;
  //criar a pilha
  iniciaPilha(s);
  //Verificar que a pilha está vazia
  if(pilhaVazia(s))
    cout << "A pilha está vazia." << endl;
  else
    cout << "A pilha não está vazia." << endl;

  //empilhar quatro elementos
  empilha(s,12);
  empilha(s,33);
  empilha(s,7);
  empilha(s,11);

  //Verificar que a pilha está cheia
  if(pilhaCheia(s))
    cout << "A pilha está cheia." << endl;
  else
    cout << "A pilha não está cheia.\n" << endl;
  //desempilhar e mostrar um elemento
  cout << "Item desempilhado: " << desempilha(s)
<< endl;
  //terminar

  system("PAUSE");
  return 0;
}
```

4 EXERCÍCIOS PROPOSTOS

a) Implemente uma pilha que armazene palavras ao invés de números.
b) Implemente uma pilha que armazene 20 teclas pressionadas pelo usuário e depois imprima o que foi digitado.

5 TERMINAMOS

Terminamos por aqui. Saia do Dev-C++ e corra para o próximo tutorial.