C++ Stak programmering

Denne artikel viser hvordan man programmere en stak i c++, som kan gemme tal værdier. Nem og hurtigt at forstå :=)

Skrevet den **04. Feb 2009** af **danielhep** I kategorien **Programmering / C/C++** | ★☆☆☆☆

C programmeret Stak

En stak er at allokere noget hukommelse, statisk i dette eksempel, hvor man så kan gemme data.
I dette eksempel gemmes der (integer) tal i stakken.

1). Vi starter med at inkludere vores to header filer med de nødvendige funktioner til vores stak.

```
1).
#include <iostream.h>
#include <conio.h>
```

2). Her defineres der hvor mange elementer vores stak maksimalt kan indeholde.

```
2).
#define MAX 25
```

3). Vores klasse til at holde de forskellige stak funktioner og arbejdes variabler.

Variabler:
int tal[MAX]; | Array som holder stakkens tal værdier
int naeste; | Tæller til funktionerne push, pop, peek, erTom

Funktioner:
Stak(void){naeste=0;} | Class constructor, initilisere variablen naeste til 0
void push(int i); | Indsætter en nyt tal i vores stak
int pop(void); | Funktion til at udskrive til fra stakken i omvendt rækkefølge
void peek(void); | Udskriver det sidste indskrevet tal fra stakken
int erTom(); | Funktion til at afgøre om stakken er tom

```
```

```
3).
class Stak
{
  private:
    int tal[MAX];
    int naeste;
  public:
    Stak(void){naeste=0;}
    void push(int i);
    int pop(void);
    void peek(void);
    int erTom();
};
```

4). Push funktionen tager et parameter som INT (heltal), og indsætter det i stakken medmindre stakken er fuld.

```
4).
void Stak::push(int i)
{
  if(naeste < MAX)
  {
    tal[naeste++] = i;
  }
  else
  {
    cout << endl << "Stakken er fuld.";
  }
}
```

5). Pop funktionen udskriver det sidste tal fra stakken, kalder du funktionen i et loop, får du hele stakken udskrevet.

```
5).
int Stak::pop(void)
{
  if(naeste > 0)
  {
    return tal[--naeste];
  }
  else
  {
```

```
    cout << "Stakken er tom.";
  }
  return 0;
}
```

6). Peek funktionen udskriver altid kun det allersidste tal som er bleven indstat i stakken.
Udføre du funktionen i et loop, får du det samme tal udskreven alt for mange gange :)

```
6).
void Stak::peek(void)
{
  if(naeste > 0)
  {
    cout << tal[naeste -1];
  }
  else
  {
    cout << "Stakken er tom.";
  }
}
```

7). erTom funktionen afgøre om stakken er tom.

```
7).
int Stak::erTom(void)
{
  return naeste == 0;
}
```

8). Main funktionen kalder vores klasse og bruger de ovenstående.

```
8).
void main(void)
{
  Stak st;
```

```
  char ch;
  cout << endl << "Skriv en række cifre: ";
  while(1)
  {
    if((ch = getche()) == '\r')
    {
      break;
    }
    else
    {
      st.push((int) ch -48);
    }
  }
  cout << endl << "Det sidste indtastede ciffer er: ";
  st.peek();
  cout << endl << "Her er alle cifrene i omvendt rækkefølge: ";
  while(!st.erTom())
  {
    cout << st.pop();
  }
  getch();
}
```

9). kopierer du det ovenstående og sammensætter det til et program og indtaster '123' som input får du følge udskrift.

```
9).

Skriv en række cifre: 123
Det sidte indtastede cifre er: 3
Her er alle cifrene i omvendt rækkefølge: 321
```

Kommentar af krismort d. 14. Nov 2004 | 1

host "templates" *host*

Kommentar af casperwollesen d. 27. Sep 2004 | 2

Kommentar af shjorth d. 04. Oct 2004 | 3

Kommentar af brilleaben d. 13. Jan 2005 | 4

Badr

Kommentar af jih d. 21. Jul 2008 | 5

Kommentar af nicklasw d. 09. Jun 2008 | 6