Cvičení #7
Dynamická alokace paměti a memory leaky

Dynamická vs statická alokace

- Statickou alokaci objektu známe:

```
string nejakyText(“Nejaky text...”);
```

- Dynamická verze vyžaduje ukazatel:

```
string *jinyText
            = new string(“Jiny text...”);
```

- Pro pole je to podobné

```
int poleCisel[20];
int *jinePoleCisel = new int[20];
```

Dealokace

- Co jsme si vzali musíme jednou vrátit

```
int *pole = new int[20];
```

…

```
delete [] pole;
```

- Nelze dealokovat dvakrát jeden ukazatel
- Nelze dealokovat nulový ukazatel

Dynamická alokace v objektech

- Doporučení pro objekty
  - Pokud používáme proměnnou typu ukazatel, je dobré ji v konstruktoru buď inicializovat nebo nastavit na NULL
  - Kdykoliv někde v objektu alokujeme paměť, rovnou provedeme i její dealokaci v destruktoru
  - Před dealokací paměti je dobré ukazatel otestovat, zda není NULL

Memory leak

- Pokud dynamicky alokujeme paměť a pokud ztratíme i její adresu, pak nelze tuto paměť dealokovat!
- Ukázka (viz Ukazka7.1.cpp) demonstruje ztrátu 4kB dat:

```
int *pole = new int[1024];

pole = NULL;
```

- Pokud aplikace skončí dříve, než dealokuje všechnu dynamicky alokovanou paměť, považujeme to také za chybu

Detekce pomocí valgrindu

```
==14248== Memcheck, a memory error detector
==14248== Copyright (C) 2002-2009, and GNU GPL'd, by Julian Seward et al.
==14248== Using Valgrind-3.6.0.SVN-Debian and LibVEX; rerun with -h for
copyright info
==14248== Command: ./Ukazka7.1
==14248==
==14248==
==14248== HEAP SUMMARY:
==14248==     in use at exit: 4,096 bytes in 1 blocks
==14248==   total heap usage: 1 allocs, 0 frees, 80 bytes allocated
==14248==
==14248== LEAK SUMMARY:
==14248==    definitely lost: 4,096 bytes in 1 blocks
==14248==    indirectly lost: 0 bytes in 0 blocks
==14248==      possibly lost: 0 bytes in 0 blocks
==14248==    still reachable: 0 bytes in 0 blocks
==14248==         suppressed: 0 bytes in 0 blocks
==14248== Rerun with --leak-check=full to see details of leaked memory
==14248==
==14248== For counts of detected and suppressed errors, rerun with: -v
==14248== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 4 from 4)
```