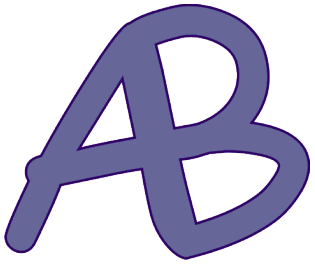

Wykład 5: Wskaźniki i zmienne dynamiczne

Wskaźniki

Definiowanie wskaźników

typ_wskazywanego_obiektu * nazwa wskaźnika;

Np.:

```
int *wsk_na_int;
char * wsk_na_znak;
float  * wsk_na_float;
```

Posługiwanie się wskaźnikami

Np.:

```
int *x;
// definicja wskaźnika do obiektów typu int
int st = 100;
// definicja obiektu typu int zliczbą 100
x = &st;
// ustawienie wskaźnika na obiekt st
cout << *x;
// wypisanie wartości obiektu wskazywanego przez x
cin >> *x; // to samo, co cin >> st;
```

Posługiwanie się wskaźnikami

Np.:

```
int *x;
// definicja wskaźnika do obiektów typu int
int st = 100;
// definicja obiektu typu int zliczbą 100
x = &st;
// ustawienie wskaźnika na obiekt st
cout << *x;
// wypisanie wartości obiektu wskazywanego przez x
cin >> *x; // to samo, co cin >> st;
```

Zmienne dynamiczne

Zmienne dynamiczne

Wskaźniki można zastosować do dynamicznej alokacji zmiennych – czyli rezerwacji w pamięci obszarów do przechowywania zmiennych.

Tak stworzona zmienna nie ma nazwy, lecz tylko adres. Adres ten przechowywany jest w statycznej zmiennej wskaźnikowej (lub bardziej skomplikowanej strukturze danych takiej jak lista lub drzewo binarne).

Zmienne dynamiczne

Do dynamicznej alokacji zmiennych służy operator new

```
int *wsk;
wsk = new int;
```

Lub krócej:

```
int *wsk = new int;
```

Do usunięcia z pamięci zmiennej dynamicznej służy operator delete

```
delete wsk;
```

Zmienne dynamiczne

Za pomocą operatora delete kasuje się tylko obiekty stworzone operatorem new

Próba skasowania czegokolwiek innego jest błędem.

Uwaga: nie należy dwukrotnie kasować obiektu.

Wyjątkiem jest zastosowanie operatora delete w stosunku do wskaźnika pokazującego na adres zerowy (NULL) – ponieważ żaden obiekt nie może mieć adresu 0 taka konstrukcja nie powoduje błędu.

Zmienne dynamiczne

Często stosowana konstrukcja zabezpieczająca przed podwójnym kasowaniem obiektów:

```
int *wskaznik new int;
// ……..
delete wskaznik;
wskaznik = NULL;
// …….
delete wskaźnik;  // nie spowoduje bledu
```

Zmienne dynamiczne

Cechy obiektów dynamicznych.

- Obiekty utworzone dynamicznie istnieją od momentu, gdy je utworzymy operatorem new do momentu, gdy je skasujemy operatorem delete.
- Obiekt utworzony dynamicznie nie ma nazwy. Można nim operować tylko za pomocą wskaźników.
- Obiektów takich nie obowiązują zwykłe zasady o zakresie ważności (zasady mówiące w których miejscach programu są widzialne, a w których niewidzialne pomimo, że istnieją). Jeśli tylko jest w danym momencie dostępny choćby jeden wskaźnik, który na taki obiekt pokazuje, to mamy do tego obiektu dostęp.
- Obiekty dynamicznie nie są inicjalizowane zerami (po utworzeniu zawierają przypadkowe wartości).

Tablice dynamiczne

Dynamiczna alokacja tablic:

```
int *wsk;
wsk = new int [1000];
```

Lub krócej:

```
int *wsk = new int [1000];
```

Rozmiar tablicy nie musi być stałą. Wystarczy, że jego wartość będzie znana w momencie alokacji tablicy (niekoniecznie w momencie kompilacji programu)

Tablice dynamiczne

Do usunięcia z pamięci dynamicznej tablicy służy konstrukcja:

```
delete  [] wsk;
```

Tablice wskaźników

W tablicy przechowywać można wszystkie rodzaje zmiennych prostych, w tym także wskaźniki adresami różnych miejsc w pamięci.

```
int  *tabl_wsk[1000];
```

Literatura:

W prezentacji wykorzystano przykłady i fragmenty:

- Gręboszj J. : ***Symfonia C++, Programowanie w języku C++ orientowane obiektowo***, Wydawnictwo Edition 2000.
- Jakubczyk K.: *Turbo Pascal i Borland C++ Przykłady*, Helion.

Warto zajrzeć także do:

- Sokół R. : ***Microsoft Visual Studio 2012 Programowanie w Ci C++,*** Helion.
- Kerninghan B. W., Ritchie D. M.: ***język ANSI C***, Wydawnictwo Naukowo Techniczne.

Dla bardziej zaawansowanych:

- Gręboszj J. : ***Pasja C++,*** Wydawnictwo Edition 2000.
- Meyers S.: ***język C++ bardziej efektywnie***, Wydawnictwo Naukowo Techniczne