XV. Wskaźniki

15.1. Odczytywanie adresu pamięci istniejących zmiennych

Język C++ w bardzo łatwy sposób umożliwia nam pobieranie adresu pamięci wybranych zmiennych. Wskaźnik zajmuje zazwyczaj 4 bajty bez względu na jaki typ danych wskazuje. Rozmiar wskaźnika może być jednak różny w zależności od użytego kompilatora (np. gdy użyjemy 64 bitowego kompilatora). Wskaźnik zwraca adres pierwszego bajta danych wybranej zmiennej. Aby pobrać adres dowolnej zmiennej wystarczy napisać: **&nazwa_zmiennej**.

```
#include <iostream>
#include <conio.h>
using namespace std;
int main()
{
  int zmienna1=213;
  int tablica[]={1,2,3,4,5,6,7,8,9,10};
  struct
  {
    int liczba;
    long long duzaLiczba;
  }struktura;
  cout<<"Adres zmienna1="<<&zmienna1<<endl<<endl;
  cout<<"Adres tablica="<<&tablica<<endl;
  cout<<"Adres tablica[0]="<<&tablica[0]<<endl;
  cout<<"Adres tablica[1]="<<&tablica[1]<<endl<<endl;
  cout<<"Adres struktura="<<&struktura<<endl;
  cout<<"Adres struktura.liczba="<<&(struktura.liczba)<<endl;
  cout<<"Adres struktura.duzaLiczba="<<&(struktura.duzaLiczba)<<endl;
  getch();
  return(0);
}
```

Zauważmy, że wskaźnik ze zmiennej **tablica** i ze zmiennej **tablica[0]** jest taki sam. Dzieje się tak dlatego, że wskaźnik ze zmiennej **tablica** wskazuje na początek wszystkich danych w tablicy, a pierwszym elementem jest **tablica[0]**. To samo dotyczy adresu zmiennej **struktura** i **struktura.liczba**. Adresy zmiennych są wyświetlane w postaci szesnastkowej.

15.2 Wskaźniki pierwsze spojrzenie.

```
//Wskaźniki pierwsze spojrzenie----------
#include <iostream>
#include <conio.h>
int main()
{
  using namespace std;

  int liczba = 9;
  int *wsk_liczba; /*--deklaracja wskaźnika
        na int--*/
  wsk_liczba = &liczba; /* przypisanie wskaźnikowi
                adresu int */
  //dwa sposoby wyświetlenia wartości liczba
  cout << "Zmienna liczba = " << liczba
  << " natomiast, *wsk_liczba = " << *wsk_liczba
```

```
  << endl;
  //dwa sposoby wyświetlenia adresu zmiennej
  cout << "Adres liczby = " << &liczba
  << " natomiast, *wsk_liczba = " << wsk_liczba
  << endl;
  //zmiana wartości za pomocą wskaźnika
  *wsk_liczba = *wsk_liczba + 1;
  cout << "Liczba = " << liczba;

  getch();
  return 0;
}
//--------------------------------------
```

Deklaracja zmiennej wskaźnikowej jest również prosta. Aby utworzyć zmienną wskaźnikową, to po typie zmiennej dopisujemy *****(gwiazdkę). Tak więc, jeśli chcemy utworzyć wskaźnik, który ma wskazywać na liczbę typu int, zapis ten będzie wyglądał tak:

Stary zapis C
// int *wsk_liczba;
Nowy zapis C++
//int* liczba, liczba1

gdzie dokładnie znajduje się *****(gwiazdka) dla kompilatora nie ma znaczenia można nawet zapisać w ten sposób:

//int * liczba

Jakiego Ty będziesz używał zapisu zależy od Ciebie, uważaj jednak na taki zapis:

//int* liczba, liczba1

w ten sposób deklarujesz jeden wskaźnik *liczba* i jeden zmiennej typu **int** *liczba1*. **Ważne !!!** zmienna, która ma być wskaźnikiem musi zawierać **gwiazdkę**.

15.2.1 Wyświetlanie adresu wskaźnika

Jeśli wypiszemy teraz wartość zmiennej wskaźnik, otrzymamy liczbę wyświetloną szesnastkowo.

```
#include <iostream>
#include <conio.h>
using namespace std;
int main()
{
  long long zmienna=213;
  long long* wskaznik=&zmienna;
  cout<<"&zmienna="<<&zmienna<<endl;
  cout<<"wskaznik="<<wskaznik<<endl;
  getch();
  return(0);
}
```

Jak pokazuje ten przykład i jak można było się tego spodziewać, wartość wskaźnika jest taka, jaką do niego zapisaliśmy.

15.2.2 Wyświetlanie danych, na które wskazuje adres wskaźnika

Aby wyświetlić dane jakie znajdują się pod adresem jaki mamy zapisany we wskaźniku, musimy przed nazwą zmiennej dopisać ***. Tak więc, modyfikując poprzedni program, będzie to wyglądało tak:

```
#include <iostream>
#include <conio.h>
using namespace std;
int main()
{
  long long zmienna=213;
  long long* wskaznik=&zmienna;
  cout<<"zmienna="<<zmienna<<endl;
  cout<<"*wskaznik="<<*wskaznik<<endl;
  getch();
  return(0);
}
```

15.3. Modyfikacja danych, na które wskazuje wskaźnik

Mając zapisany adres do zmiennej we wskaźniku, mamy możliwość zmiany wartości zmiennej nie używając nazwy zmiennej, z której pobraliśmy adres. Przykład:

```
#include <iostream>
#include <conio.h>
using namespace std;
int main()
{
  long long zmienna=213;
  long long* wskaznik=&zmienna;
  cout<<"zmienna="<<zmienna<<endl;
  *wskaznik=50;
  cout<<"zmienna="<<zmienna<<endl;
  getch();
  return(0);
}
```

15.4. Bezpieczeństwo a użycie wskaźników.

Podczas używania wskaźników można popełnić błąd przy tworzeniu i używaniu wskaźnika. Jeżeli nie zadbamy o przypisanie wskaźnikowi adresu.

```
int *wsk_liczba;
*wsk_liczba = 373;
```

Jest to błąd, ponieważ nie wiemy gdzie(pod jakim adresem) umieszczona jest wartość 373. Kompilator będzie starał się umieścić wartość 373 bo takim właśnie adresem. Co jeżeli jednak adres ten będzie już zajęty przez program? Taka sytuacja może spowoduje, iż nie będzie można było zapisać nic w miejsce wskazane przez wsk_liczba. Błąd ten jest nie dopuszczalny i do tego trudno go wykryć. **Pamiętaj gdy używasz wskaźnika zadbaj o to by zawsze miał przypisaną zmienną(miał prawidłowy adres).**

15.5. Arytmetyka wskaźników.

Wskaźniki posiadają pewne podobieństwo z nazwami tablic. Wywodzi się to z arytmetyki wskaźników i sposobie przedstawienia tablic w C++. Przykład pokazuje wspomnianą analogię:

```
//Wskaźniki drugie starcie----------
#include <iostream>
#include <conio.h>
int main()
{
 using namespace std;
 //tablice deklaracia inicjalizacja
 double waga[5] = {55.3, 747.8, 1001.2, 5.2, 6.4};
 short odliczanie[4] = {3, 2, 1, 0};
 //wskaźniki
 double *wsk_waga = waga; //nazwa tabeli = adres
 short *wsk_odliczanie = &odliczanie[0];

 //Wyświetlanie adresu i wartości wskaźnika wsk_waga
 cout << "wsk_waga = " << wsk_waga
 << ", *wsk_waga = " << *wsk_waga
 << endl
 << "Dodawanie wsk_waga + 1 ";
 wsk_waga += 1;
 cout << "\nTeraz wsk_waga = " << wsk_waga
 << ", *wsk_waga = " << *wsk_waga << endl
 << endl;
 //Wyświetlanie adresu i wartości wskaźnika wsk_odliczanie
 cout << "wsk_odliczanie = " << wsk_odliczanie
 << ", *wsk_odliczanie = " << *wsk_odliczanie
 << endl
 << "Dodawanie wsk_odliczanie + 1 ";
 wsk_odliczanie += 1;
 cout << "\nTeraz wsk_odliczanie = " << wsk_odliczanie
 << ", *wsk_odliczanie = " << *wsk_odliczanie << endl
 << endl;

 //Wyświetlanie zapisu tablicowego
 cout << "\nPodobienstwa tablic i wskaznikow\n"
 << "Pierwszy element tab waga[0] = "
 << waga[0] << endl
 << "Drugi element tab odliczanie[1] = "
 << odliczanie[1] << endl << endl;

 //Wyświetlanie zapisu wskaźnikowego
 cout << "Pierwszy element tab waga "
 "z uzyciem wskaznika *waga = "
 << *waga << endl
 << "Drugi element tab odliczanie "
 "z uzyciem wskaznika *(odliczanie + 1) = "
 << *(odliczanie + 1) << endl << endl << endl;

 //porównanie wielkości tablic i wskaźników
 cout << "Tablica waga wazy " << sizeof(waga)
 << " bajtow!" << endl
 << "Jednak wskaznik na ta tablice *wsk_waga "
 << "wazy tylko " << sizeof(wsk_waga)
 << " bajty!" << endl;

 getch();
 return 0;
```

```
}
//--------------------------------------
```

Pierwsza część programu wyświetlenia adresy wskaźników ***wsk_waga** i ***wsk_odliczanie**, oraz wartości zapisane pod adresami na które wskazują. Następnie dodajemy 1 do obu wskaźników, co powoduje przesunięcie ich adresów o **8 bajtów** dla wsk_waga(ponieważ typ double to 8 bajtów) i **2 bajty** dla wsk_odliczanie(typ short to 2 bajty). Przesunięcie powoduje, iż oba wskaźniki wskazują na adres drugiej wartość w obu tablicach. **Wniosek** dodanie do wskaźnika **1** powoduje jego **przesunięcie o tyle bajtów** ile ma wskazany typ danych.

Kolejna część programu pokazuje zależności między tablicą a wskaźnikiem. Dlaczego zapisy **waga[0] = *waga**, oraz **odliczanie[1] = *(odliczanie + 1)** dają ten sam rezultat? Ponieważ tak działa kompilator C++, zamienia on zapis **tab[10]** dla bardziej czytelny sobie zapis ***(tab + 10).** Stąd właśnie wskaźniki i nazwy tablic można używać zamiennie. Różnice między **tab., a wsk.** to:

NazwaWskaźnik = NazwaWskaźnika + 1; //**prawidłowo**
NazwaTabel = NazwaTabeli + 1; // **błąd !!!**

oraz różnią się wielkością, którą możesz sprawdzić sam stosując operator **sizeof**.

15.6. Dostęp do danych struktury za pośrednictwem wskaźnika

Jeśli chcemy odczytać lub zapisać dane do struktury za pomocą wskaźnika wskazującego na nią, postępujemy prawie tak samo jak w przypadku zwykłej zmiennej - poprzedzamy wskaźnik znakiem ***. Wskaźnik ten musimy jednak umieścić w okrągłe nawiasy, żeby kompilator wiedział czego się tyczy symbol ***. Kolejny przykład:

```
#include <iostream>
#include <conio.h>
using namespace std;
int main()
{
  struct daneST
  {
    int liczba;
    char znak;
  };
  daneST dane;
  dane.liczba=55;
  dane.znak='a';
  daneST* wskaznik=&dane;
  cout<<"(*wskaznik).liczba="<<(*wskaznik).liczba<<endl;
  (*wskaznik).liczba=99;
  cout<<"dane.liczba="<<dane.liczba<<endl;
  getch();
  return(0);
}
```

15.7. Wskaźniki i struktury po raz drugi

Oprócz przedstawionej wyżej metody uzyskiwania dostępu do danych istnieje również drugi, który jest równoważny pierwszemu. Jest on moim zdaniem wygodniejszy w użyciu, jednak chciałem pokazać Ci różne zapisy ponieważ starsi programiści, którzy 'przesiedli' się z C na C++ korzystają zazwyczaj z pierwszego zapisu. Zamiast poprzedzać zmienną wskaźnikową gwiazdką i wstawiać ją w nawiasy, wystarczy kropkę zastąpić zapisem takim zapisem: ->. Przykład z poprzedniego podrozdziału ze zmodyfikowanym zapisem przedstawiam poniżej.

```
#include <iostream>
#include <conio.h>
using namespace std;
int main()
{
  struct daneST
  {
    int liczba;
    char znak;
  };
  daneST dane;
  dane.liczba=344;
  dane.znak='a';
  daneST* wskaznik=&dane;
  cout<<"wskaznik->liczba="<<wskaznik->liczba<<endl;
  wskaznik->liczba=221;
  cout<<"dane.liczba="<<dane.liczba<<endl;
  getch();
  return(0);
}
```

15.8. Podsumowanie

Przeanalizuj dokładnie cały materiał, jaki znalazł się w tym rozdziale. Dobra znajomość całej teorii o wskaźnikach będzie niezbędna, gdy dojdziesz do rozdziału poświęconemu dynamicznemu zarządzaniu pamięcią.

15.9 Ćwiczenia

1. Popraw błędy w następującym kodzie:

```
//Wskaźniki pierwsze zadanie----------
#include <iostream>
#include <conio.h>
int main()
{
  using namespace std;

  short zmienna = 213;
  short long* wskaznik = zmienna;

  //Wyświetlanie adresu wskaźnika
  cout << "&zmienna=" << zmienna << endl
  cout <<"wskaznik=" << wskaznik << endl;
  //Wyświetlanie danych, na które wskazuje adres wskaźnika
  cout  "Adres zmienna=" << *zmienna <<endl;
  cout << "*wskaznik=" << wskaznik
  //Modyfikacja danych, na które wskazuje wskaźnik
  cout << "zmienna="<<zmiena<<endl;
  *wskaznik = &50;
  cout << "zmienna=" << zienna <<endl;

  getch();
  return 0;
}
//---------------------------------------
```