

# Wskaźniki

## Informatyka

### Materiały

Wskaźnik z punktu widzenia programisty jest grupą komórek pamięci (rozmiar wskaźnika zależy od architektury procesora, najczęściej są to dwa lub cztery bajty ), które mogą zmieścić adres początkowy pewnego obiektu. Jeżeli *str* jest obiektem typu **char**, a *pstr* ma być wskaźnikiem, który wskazuje na ten obiekt, to taki wskaźnik można zdefiniować następujący sposób:

```
char str;  
char *pstr;  
pstr=&str;
```

### Deklaracja

```
typ *wskaznik;  
typ tablica[rozmiar];
```

Zarówno *wskaznik* jak i *tablica* są zmiennymi typu wskaźnikowego (przy czym *tablica* jest zmienną, której nie można przypisać innej wartości).

### Przypisanie adresu do wskaźnika

```
wskaznik = inny_wskaznik;  
wskaznik = &zmienna;
```

### Odczyt wartości z adresu / zapis pod adresem

```
zmienna = *wskaznik;  
*wskaznik = wartosc; // stała, zmienna, funkcja lub wyrażenie
```

### Działania na wskaźnikach

```
wskaznik++;  
wskaznik--;  
wskaznik + indeks;  
wskaznik[indeks];
```

## Uwagi, komentarze

Operator wyluskania (*\*zmienna*) ma wysoki priorytet, szczególnie z operatorami ++ i -- należy być ostrożnym (czy to dotyczą wskaźnika czy też wartości w pamięci pod adresem wewnątrz wskaźnika)

Powyższe działania można łączyć:

```
wskaznik = inny_wskaznik + indeks;  
zmienna = *(wskaznik + indeks);  
zmienna = *wskaznik++;
```

## Dynamiczna alokacja pamięci

- Czasami z góry nie wiadomo ilu elementowa tablica będzie potrzebna. Deklaruje się wówczas zmienną wskaźnikową pożądanego typu i wywołuje funkcje alokacji pamięci (o żądanym rozmiarze). Funkcja ta zwraca adres pamięci. Należy to wpisać do wskaźnika na którym można później już normalnie działać (np. `wskaznik[indeks] = wartosc;`).
- Kiedy zaalokowana pamięć jest niepotrzebna, należy ją zwolnić.
- Uwaga - niezwolnione obszary pamięci, czytanie i zapisywanie pod niedozwolone adresy to częste i poważne błędy programistyczne.
- Funkcje alokujące i zwalniające pamięć `malloc()` i `free()` w C zostały zastąpione w C++ przez operatory **new** oraz **delete**.

```
wsk1 = new typ;  
wsk2 = new typ[ ilosc_elementow];  
delete wsk;  
delete[] wsk2;
```

- Obszar pamięci jaki zostanie przydzielony może zawierać dowolne wartości ("śmieci", poprzednie dane). Nie wolno robić żadnych założeń co do zawartości pamięci.

## Przykład

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]){
    int *wsk; int i, n;

    printf("Podaj rozmiar tablicy");
    scanf("%d", &n);

    wsk = new int[n]; // zaalokuj pamięć

    for (i = 0; i < n; ++i) wsk[i] = i;//wypełnij tablicę

    delete [] wsk; //zwolnij pamięć
    return 0;
}
```

## Wskaźniki typu *void*

W C wskaźniki dowolnego typu mogą być przydzielane do wskaźników typu *void* i odwrotnie, natomiast w C++ wymagana jest jawna konwersja.

```
void *wsk;
char *str ="Tekst";
wsk = (char*) str;
```

## Referencja

W C++ można tworzyć tzw. zmienne referencyjne. Jeśli zadeklarujemy zmienną referencyjną do innej zmiennej, to jakiegokolwiek zmiany wartości dowolnej z powiązanych w ten sposób zmiennych będą odnosiły się również do drugiej z nich.

```
int w = 4;
int &n = w; //n jest zmienną referencyjną do w
n = 12; //w == 12 && n == 12
w = 8; //w == 8 && n == 8
```

Referencja jest najczęściej wykorzystywana w funkcjach modyfikujących swoje argumenty. (W C konieczne było w tym celu przekazywanie do funkcji wskaźników do zmiennych)

```
void change(int &a, int &b){
    int t;
    t = a;
    a = b;
    b = t;
}
```

```
int main(){
    int *wsk; int a = 3, b = 5;
    change(a, b);
    //a == 5 && b == 3
    return 0;
}
```

W C++ operatory *new* i *delete* (W C *malloc*, *calloc*, *realloc*, *free*, *sizeof*, konwersje wskaźników z *void\**)

```
char *str, *str1;
str = new char;
*str = 'a';
str1 = new char ('b');
char *str3;
str3 = new char[10];

if (str3) strcpy(str3, "1234567890");
else printf("Błąd przydziału pamięci\n");

printf("str = %s\n", str3);
printf("str = %c\n", *str);
```

- Obiekty utworzone za pomocą **new** istnieją aż do momentu ich jawnego usunięcia operatorem **delete**
- Po utworzeniu zawierają "śmieci"

```
delete str1;
//usunięcie tablicy dynamicznej
```

```
delete[] str2;
```

## Pozostałe informacje

- W C zmienna typu char to przede wszystkim liczba 8-bitowa. Aby w kodzie wykorzystywać stałe literały znakowe (np. literę A) należy je umieścić pomiędzy apostrofami (np. 'A').
- Czymś innym jest 0, 1, 2, ... niż '0', '1', '2', ...
- Wypisanie zmiennej typu char na konsolę (innymi słowy, instrukcja dla kompilatora by liczbę 8-bitową potraktował jako czytelny znak) to printf("%c", zmienna);. Uwaga, czytelne znaki nie zajmują całej przestrzeni 0-255, część wartości koduje znaki sterujące (np. tabulator) i inne.
- W C napisy to zmienne typu char \*s, czyli dla programistów - tablice zawierające wartości typu char zakończone pojedynczym bajtem 0. Długość napisu to zatem różnica adresów bajtu 0 i początku napisu.