

Podstawy języka C – cz. II

dr inż. Maciej Kusy

Katedra Podstaw Elektroniki
Wydział Elektrotechniki i Informatyki
Politechnika Rzeszowska

Elektronika i Telekomunikacja, sem. 2

Plan wykładu

- Typy pochodne:
 - wskaźniki
 - tablice
- Pojęcie funkcji
 - deklaracja i definicja funkcji
 - wywołanie funkcji
 - argumenty funkcji
 - zmienne lokalne i globalne



Typy pochodne – wskaźniki

Wskaźnik rodzaj zmiennej, który wskazuje na miejsce w pamięci, gdzie zapisana jest dana informacja.

Aby wykorzystywać wskaźnik, musi on wskazywać na określony typ (*) i musi mu być przypisany adres zmiennej, której typ jest zgodny z typem wskaźnika.

```
int i = 5, j = 6, k;  
int* p;  
p = &i;    // => *p => 5, p => 0021FAAC (np.)  
*p = 2;    // => i = 2; (modyfikacja i)  
p = &j;  
*p *= 2;   // => j *= 2; j = 12;  
k = *p;    // => k = 12;
```

Właściwości wskaźników

- Wskaźnik wskazuje na obszar pamięci, w którym zapisana jest zmienna jakiegoś typu, struktury;
- Wskaźnik jest tego samego typu co zmienna, na którą pokazuje;
- Wskaźnik deklaruje się poprzez operator ***** przy typie zmiennej, np: **int *p;**
- Modyfikacja wartości pokazywanej przez wskaźnik (***p**) spowoduje modyfikację zmiennej, na którą pokazuje;
- Aby dostać się do wartości wskazywanej przez wskaźnik, należy użyć operatora wyłuskania: *****.

Typy pochodne – tablice

Tablica to kolekcja, która przechowuje dane tego samego typu w zorganizowany sposób.

Matematycznym odpowiednikiem tablicy jest ciąg.

Dostęp do poszczególnego elementu tablicy uzyskuje się poprzez odwołanie się do nazwy tablicy i odpowiedniej wartości indeksu ($0, \dots, n-1$) umieszczonego w `[]`.

Nazwa tablicy jest jednocześnie jej adresem w pamięci.

Rozmiar tablicy może być ustalony z góry (statycznie), ale nie w trakcie wykonywania programu.

Deklaracja tablic

Tablicę definiuje się za pomocą następującej składni:

Tablica statyczna na 10 elementów:

```
typ nazwa[10];
```

```
int tab[10]={1,2,3,4,5,6,7,8,9,10};
```

```
tab[0] = 1;
```

```
tab[9] = 10;
```

```
double liczby[] = {1.1,2.2,3.3,4.4,5.5};
```

```
char test[80];
```

```
char test[] = "Programowanie w j. C";
```

```
//test[0]='P'   test[1]='r'   test[20]='\0'
```

Właściwości tablic

- Tablice definiuje się dla danego typu;
- Długość tablicy może przyjmować wartość stałą, np.:
char tablica[5];
- Tablice mogą być tworzone dynamicznie za pomocą wskaźników;
- Nazwa tablicy jest wskazaniem na zerowy jej element;
- Indeksacja elementów rozpoczyna się od **0**;
- Ostatnim elementem n -elementowej tablicy jest element o indeksie **$n-1$** ;
- Odniesienie się do danego elementu tablicy polega na podaniu jej nazwy oraz odpowiedniego indeksu w nawiasach **[]**.

Wskaźniki i tablice


Wskaźniki można ustawić tak, aby pokazywały na tablicę.

```
int tab[5] = {1, 2, 3, 4, 5};
int* p;
p = &tab[0]; // p = tab;


*p => 1


```

```
* (p + 2) = 0;
p[2] = 0;
```



```
tab[2] = 0
```

```
tab[0] = 1, tab[1] = 2, tab[2] = 0,
tab[3] = 4, tab[4] = 5
```


Funkcje w języku C

Funkcja – procedura implementująca określone działanie. Funkcja definiowana jest w wydzielonej części programu. Posiada: typ zwracany, nazwę, listę argumentów ujętych w nawiasy okrągłe oraz ciało umieszczone w nawiasach klamrowych.

Sposób definiowania funkcji w języku C:

```
typ nazwa(typ arg_1, typ arg_2, ...)  
{  
    //instrukcje  
}
```

Deklaracja funkcji

Deklaracja funkcji – podanie kompilatorowi informacji na temat „szczegółów” funkcji (typ zwracany, nazwa, argumenty).

Po umieszczeniu deklaracji zapowiadającej funkcji, przy każdorazowym użyciu funkcji w dalszej części programu kompilator weryfikuje, czy wywołanie funkcji jest prawidłowe oraz z którą funkcją ma do czynienia.

Przykładowa deklaracja zapowiadająca funkcji:

```
double Srednia(double* tab, int n);
```

Przy deklaracji nie jest konieczne podanie nazw argumentów:

```
double Srednia(double*, int);
```

Definicja funkcji

Definicja funkcji – jest jej deklaracją wraz z implementacją (ciało funkcji) umieszczoną w nawiasach klamrowych { }.

Deklaracji funkcji może być kilka (w projektach wieloplikowych) natomiast definicja funkcji może być tylko jedna.

```
double Srednia(double* tab, int n)
{
    double suma = 0; int i;
    for(i = 0; i < n; i++)
        suma += tab[i];
    return suma / (1.0 * n);
}
```

Wywołanie funkcji

Wywołanie funkcji – podanie w kodzie programu nazwy funkcji wraz z opcjonalnymi argumentami oddzielonymi przecinkami, które umieszczone są w nawiasach okrągłych.

Liczba argumentów podczas wywołania musi być równa liczbie parametrów zdefiniowanej funkcji.

Niepodanie `()` interpretowane jest jako adres funkcji.

```
double tab[5] = {1.1, 2.2, 3.3, 4.4, 5.5};  
printf("%f\n", Srednia(tab, 5));
```

Argumenty funkcji

Jeżeli funkcja zdefiniowana jest z listą argumentów, można ją wywołać przekazując jej odpowiednio zdefiniowane wcześniej zmienne.

Wystarczy wówczas w nawiasach okrągłych podczas wywołania podać nazwy tych zmiennych.

Kolejność i typ przesyłanych zmiennych muszą być zgodne z kolejnością i typem argumentów w definicji funkcji.

Zmienne można przekazywać do funkcji poprzez:

- wartość: praca na kopii zmiennej w funkcji,
- wskaźnik: praca na oryginale zmiennej w funkcji.

Funkcja może posiadać zmienną liczbę argumentów ustawioną podczas deklaracji:

```
void fun(char typ[], ...);
```

Zmienne lokalne i globalne

Ciało funkcji ujęte jest w zakresie nawiasów klamrowych. Przez to, wewnątrz funkcji tworzony jest zakres lokalny.

Jeżeli w zakresie lokalnym funkcji zostaną stworzone jakiekolwiek zmienne (stos) – będą to zmienne lokalne. Przestaną one istnieć po zakończeniu działania funkcji.

Jeżeli nazwa i typ zmiennej lokalnej zdefiniowanej w funkcji będzie taka sama jak nazwa i typ zmiennej o zakresie globalnym, to wewnątrz funkcji zmienna lokalna przysłoni zmienną globalną.