

Programowanie komputerowe

Zajęcia 5

Wskaźniki

- Wskaźnik to zmienna, której wartością jest miejsce w pamięci komputera, w którym przechowywana jest wartość pewnego typu.
- Typ wskaźnikowy do typu T to T^* .
- Wartością wskaźnika może być 0 (`NULL`), co oznacza, że wskaźnik nie wskazuje na nic.
- Wartość wskazywaną przez wskaźnik p uzyskujemy przez $*p$.
- Wskaźnik do zmiennej v uzyskujemy przez $\&v$.
- Zastosowanie: wskazywanie tablic i obiektów tworzonych dynamicznie.

Tablice tworzone dynamicznie

Jeśli potrzebujemy tablicy, której rozmiar nie jest znany w momencie kompilacji, możemy stworzyć ją dynamicznie. Przykład poniżej:

- Deklarujemy wskaźnik, za pomocą którego będziemy mieć dostęp do tablicy:

```
int* p;
```

- Tworzymy tablicę przy pomocy operatora `new` i ustawiamy wskaźnik na jej pierwszy element. Rozmiar może być dowolnym wyrażeniem.

```
p = new int[rozmiar];
```

- Używamy tablicy; jej poszczególne elementy to `p[0]`, `...`, `p[rozmiar-1]`
- Usuwamy tablicę przy pomocy operatora `delete`:

```
delete[] p;
```

Dostęp do tablic przy pomocy wskaźników

- Tablicę można utożsamiać ze wskaźnikiem wskazującym na jej pierwszy element.
- Dla wskaźników można używać operatorów ++, --, +, -.
- ++ (--) powoduje przesunięcie wskaźnika o jedno miejsce w prawo (lewo).
- dodanie (odjęcie) liczby całkowitej powoduje przesunięcie wskaźnika o podaną liczbę miejsc w prawo (lewo).
- można odejmować wskaźniki jeśli wskazują na tą samą tablicę.

Wskaźniki i tablice – przykład

```
int main() {  
    int t[5]={2,4,7,5,7};  
    for(int* p=t; p<t+5; p++)  
        cout << *p << endl;  
}
```

Ten program drukuje tablicę na ekranie.

Napisy w stylu C

Konwencja: zapisany w tablicy ciąg znaków zakończony jest znakiem o kodzie 0, można go zapisać jako 0, NULL lub '\0' (nie mylić ze znakiem '0').

Przykład: tablica o elementach

```
'C' 'z' 'w' 'a' 'r' 't' 'e' 'k' '\0' 'A' '?'
```

reprezentuje napis "Czwartek". Znaki znajdujące się po znaku '\0' nie są istotne.

Dzięki temu, że 0 kończy napis nie ma potrzeby przekazywania jego długości.

Konwencja ta dotyczy **tylko** tablic znaków, które reprezentują napis.

Przykłady funkcji

Funkcja zwracająca długość napisu:

```
int dlugosc(char* s) {  
    int i;  
    for(i=0; s[i]; i++)  
    return i;  
}
```

- używamy konwencji, że wyrażenie typu całkowitego lub znakowego jest prawdziwe jeśli jest różne od zera
- jeśli ciało pętli jest puste, możemy użyć średnika zamiast { }

Jeszcze raz długość napisu

```
int dlugosc(char* s) {  
    char* p=s;  
    while(*p++);  
    return p-s-1;  
}
```

Niezbyt czytelnie ale krótko. W jaki sposób działa ta funkcja?

Napisy – przykład

Funkcja sprawdzająca czy dwa napisy są jednakowe:

```
bool rowne(char* s, char* t) {  
    int i=0;  
    while(s[i]==t[i]) {  
        if(s[i]==0)  
            return true;  
        i++;  
    }  
    return false;  
}
```

Zadania

1. Napisać funkcję

```
int wystapienia(char* s, char c)
```

która zwraca liczbę wystąpień znaku `c` w napisie `s`.

2. Napisać funkcję

```
void drukujWspak(char* s)
```

która drukuje napis od ostatniego znaku do pierwszego.

3. Napisać funkcję

```
void drukujBezSpacji(char* s)
```

która drukuje na ekranie napis `s` pozbawiony spacji.

4. Napisać funkcję

```
int ileWyrazow(char* s)
```

która zwraca liczbę wyrazów w napisie `s`.

Zadania (2)

5. Napisać funkcję

```
int znajdz(char* s, char* t)
```

która zwraca indeks pierwszego wystąpienia napisu t w s (lub -1 jeśli nie ma).

6. Napisać funkcję

```
int liczba(char* s)
```

która zwraca wartość liczbową napisu s (np. liczba("376")=376).

7. Napisać funkcję

```
void drukuj(int n, int p)
```

która drukuje liczbę n w systemie pozycyjnym o podstawie p.

8. Napisać funkcję

```
bool anagramy(char* s, char* t)
```

która zwraca true jeśli s i t to anagramy.