Programowanie komputerowe

Zajęcia 4

Referencje

Jeśli `T` jest pewnym typem, to `T&` jest typem referencyjnym do `T`. Wartością typu referencyjnego jest **zmienna** typu `T`.

Typu referencyjnego możemy użyć jako parametru funkcji. Wówczas nie powstaje kopia wartości, a funkcja wywoływana używa zmiennej zadeklarowanej w innej funkcji. Robimy tak jeśli:

- chcemy zmieniać zmienne z funkcji wywołującej,
- nie chcemy (lub nie możemy) tworzyć kopii istniejącej zmiennej.

Przekazanie parametru za pomocą referencji nazywamy przekazaniem **przez zmienną**, a w zwykły sposób przekazaniem **przez wartość**.

Referencje – przykład

Poniższa funkcja zamienia wartości dwóch zmiennych:

```
void zamien(int& a, int& b) {
    int c=a;
    a=b;
    b=c;
}
```

Przykład użycia poniżej. Co się stanie jeśli pominiemy znaczki & ?

```
int main() {
    int x=3, y=7;
    zamien(x,y);
    cout << "x=" << x << ", y=" << y << endl;
}
```

Tablice

- Tablice służą do przechowywania wielu zmiennych tego samego typu.
- Tablicę deklaruje się następująco:

  ```
  typ nazwa[liczba_elementów];
  ```

  Tak zadeklarowana tablica zawiera elementy `nazwa[0]`, `nazwa[1]`, `nazwa[2]`, ..., `nazwa[liczba_elementów-1]`, wszystkie one są typu podanego w deklaracji.
- Poszczególne elementy tablicy zachowują się jak zwykłe zmienne.
- Można ustawić elementy tablicy w momencie deklaracji, np. tak:

  ```
  int t[5]={3,7,-1,2,4};
  ```

Tablice jako parametry funkcji

- Tablice są zawsze przekazywane przez zmienną – funkcja nie działa na kopii tylko na oryginalnej tablicy.
- Nie jest przekazywany rozmiar tablicy; żeby go przekazać należy użyć dodatkowego parametru, np.

```
int f(int tablica[], int rozmiar) { … }
```

- Innym sposobem jest przekazanie **wskaźnika** do tablicy, np. tak:

```
int f(int* tablica, int rozmiar) { … }
```

  O wskaźnikach – później.
- Napisy są zapamiętywane jako tablice znaków (`char`).

Ćwiczenia

Napisać poniższe funkcje. Parametr `r` oznacza długość tablicy `t`.

1. `void drukuj(int t[], int r)`
   która drukuje elementy tablicy t na ekranie.
2. `void wczytaj(int t[], int r)`
   która prosi o podanie wartości tablicy t.
3. `int max(int t[], int r)`
   która zwraca największy element tablicy t.
4. `int max2(int t[], int r)`
   która zwraca drugi największy element tablicy t.

Ćwiczenia (2)

5. `int suma(int t[], int r)`
   która zwraca sumę elementów tablicy `t`.
6. `bool rowne(int t1[], int t2[], int r)`
   która zwraca true jeśli tablice `t1` i `t2` są równe (mają takie same elementy).
7. `bool niemalejaca(int t[], int r)`
   która zwraca true jeśli `t[0]<=t[1]<= ... <=t[r-1]`.
8. `void naKoniec(int t[], int r)`
   która przestawia ostatni element tablicy `t` na koniec.
9. `void wspak(int t[], int r)`
   która zamienia kolejność elementów tablicy na odwrotną (np. `[1 6 3 2]` zamienia na `[2 3 6 1]`).

Ćwiczenia (3)

10. `int nwd(int t[], int r)`
    która zwraca największy wspólny dzielnik elementów tablicy `t`.
11. `double srednia(double t[], int n)`
    zwracającą średnią elementów tablicy.
12. `double mediana(double t[], int r)`
    zwracającą medianę elementów tablicy.
13. `double odchStd(double t[], int r)`
    zwracającą odchylenie standardowe elementów tablicy `t`.
14. `double kowariancja(double t1[], double t2[], int r)`
    zwracającą kowariancję pomiędzy elementami tablic `t1` i `t2`.