Informatyka I: Podstawy 2

Październik 2012 — Łukasz Łaniewski-Wołłk*

4 października 2012

1 Trochę więcej szczegółów

Omówmy pewne rzeczy trochę dokładniej.

1.1 Typy

W C i C++, musimy deklarować zmienne. Tzn: powiedzieć jakich będziemy używać zmiennych i jakich one będą typów. Deklaracje piszemy 'typ zmienna1,zmienna2, ...;'. Najważniejsze typy to:

- `int` — Liczba całkowita (32-bitowa, od -2^{31} do 2^{31})
- `float` — Liczba zmienno-przecinkowa. Może opisywać ułamki dziesiętne z ok 7 cyframi znaczącymi (32-bity)
- `double` — Liczba zmienno-przecinkowa. Ma 16 cyfr znaczących (64-bity)

Pamiętaj: Jeśli używasz liczb rzeczywistych (a nie całkowitych), używaj typu `double`.

Pierwszym przykładem niech będzie:

```
double a;
a = 0;
while (a < 2*3.14)
{
    circle(a * 100, sin(a) * 100 + 100, 3);
    a = a + 0.001;
}
```

Ten program narysuje wykres sinusa przeskalowany o 100, za pomocą kółek o promieniu 3.

Ćwiczenia

Używając analogicznej pętli

- Narysuj wykres a^2.
- Narysuj punkty o współrzędnych $x = 100 \sin a + 100$ i $y = 100 \cos a + 100$
- Narysuj punkty o współrzędnych $x = 100 \sin a \cos 4a + 100$ i $y = 100 \cos a \cos 4a + 100$
- Narysuj punkty o współrzędnych $x = 100r \sin a + 100$ i $y = 100r \cos a + 100$, gdzie $r = \frac{\cos a + 2}{3}$ (niech r będzie kolejną zmienną).

*llaniewski@meil.pw.edu.pl

1.2 Typy — pułapki

Ważne by pamiętać, że liczby bez przecinka dziesiętnego, są uważane za całkowite. Tzn: wykonywane są na nich działania jak dla liczb całkowitych. Dlatego `1/4` da jako wynik 0! Bo wynik 0.25 zostanie obcięty do liczby całkowitej. Żeby tego uniknąć możemy napisać `1.0/4`. Możemy także bezpośrednio 'zrzutować' zmienne z `int` na `double` pisząc: `(double) zmienna`.

Pamiętaj: Wszędzie gdzie robisz obliczenia używaj `double`. Unikaj mieszania liczb całkowitych i zmienno-przecinkowych. Nigdy nie pisz ułamków jako 1/3

Ćwiczenia

Przeanalizuj (i przetestuj) wynik tego programu — które linie nie dadzą pożądanego efektu:

```
double a;
a = 0;
while (a < 2)
{
    circle(a * 100, sin( a * 3.14 ) * 100 + 100, 3);
    circle(a * 100, sin( a * (314 / 100) ) * 100 + 100, 3);
    circle(a * 100, sin( (a * 314) / 100 ) * 100 + 100, 3);
    a = a + 0.001;
}
```

2 Funkcje po raz drugi

Zestawy operacji, które powtarzamy w programie wielokrotnie, możemy zamknąć w funkcjach. Taka funkcja „połyka" parametry i coś z nimi robi. Dla przykładu:

```
void kreski(int n, double r)
{
    int i;
    i = 0;
    while (i < n)
    {
        line(i, 0, i, r * i);
        i = i + 1;
    }
}
```

W pierwszej linii mówimy:

- jak nazywa się funkcja — `kreski`
- jakie ma parametry — `n` typu `int` i `r` typu `double`
- jakiego typu zwraca wartość —w naszym wypadku `void` oznacza, że nic nie zwraca

Funkcja ta wywołana `kreski( 20, 0.4)` wyrysuje 20 pionowych kresek o długości od 0 do $0.4 \cdot 19$ (dlaczego 19 a nie 20?)

Taką funkcję możemy wykonać wielokrotnie dla różnych wartości n i r:

```
void main()
{
    graphics(200,200);

    kreski( 10, 1.000);
    kreski( 20, 0.500);
    kreski( 30, 0.333);
    kreski( 40, 0.250);
    kreski( 50, 0.200);
    kreski( 50, 0.166);
    kreski( 70, 0.142);
    kreski( 80, 0.125);
    kreski( 90, 0.111);
    kreski(100, 0.100);

    wait();
}
```

Ćwiczenia

Napisz i wywołaj funkcje która:

- Narysuje ludzika wysokości `h` i środku głowy w (`x`, `y`)
- W pętli narysuje tłum (używając poprzedniej funkcji)
- Narysuje `n` kółek w punkcie (`x`,`y`) o coraz większych promieniach

* Narysuje wielokąt foremny o `n` bokach

3 Instrukcja warunkowa

Kolejnym bloczkiem składowym programowania, jest instrukcja warunkowa. Sprawdza ona warunek i wykonuje pewną część kodu, tylko gdy warunek jest spełniony.

```
x = 2.0;
if ( x > 0 ) {
    y = sqrt(x);
} else {
    y = 0;
}
```

Instrukcja ta sprawdza czy $x > 0$ i jeśli jest to prawdą, to wstawia $\sqrt{x}$ do zmiennej y. Gdy warunek nie jest spełniony, wykonywana jest część po `else`, więc wstawiane jest 0 do y. W ten sposób możemy zabezpieczyć się na przykład przed niemożliwymi obliczeniami, albo uzależnić działanie programu od jakiś wartości.

Zobaczmy prosty przykład:

```
double a;
a = 0;
while (a < 2*3.14)
{
    if (a < 2) {
```

```
        circle(sin(a) * 100 + 100, cos(a) * 100 + 100, 5);
    } else {
        circle(sin(a) * 100 + 100, cos(a) * 100 + 100, 10);
    }
    a = a + 0.001;
}
```

Gdyby nie instrukcja `if`, ten program narysował by koło z małych kółek. Teraz, gdy kąt `a` przekroczy 2 radiany zmieni promień kółeczka z 5 na 10

Ćwiczenia

Napisz program który:

- Dla parametru w rysuje wykres $x^2 - w$, przeskalowany o 100 w obu kierunkach i przesunięty na środek (patrz poprzedni przykład).
- Wyrysuje większe kółka w miejscach przecięcia wykresu z osią x (jeżeli przecina).
- Zmodyfikuj program by działał dla dowolnych a, b, c i funkcji $ax^2 + bx + c$.