

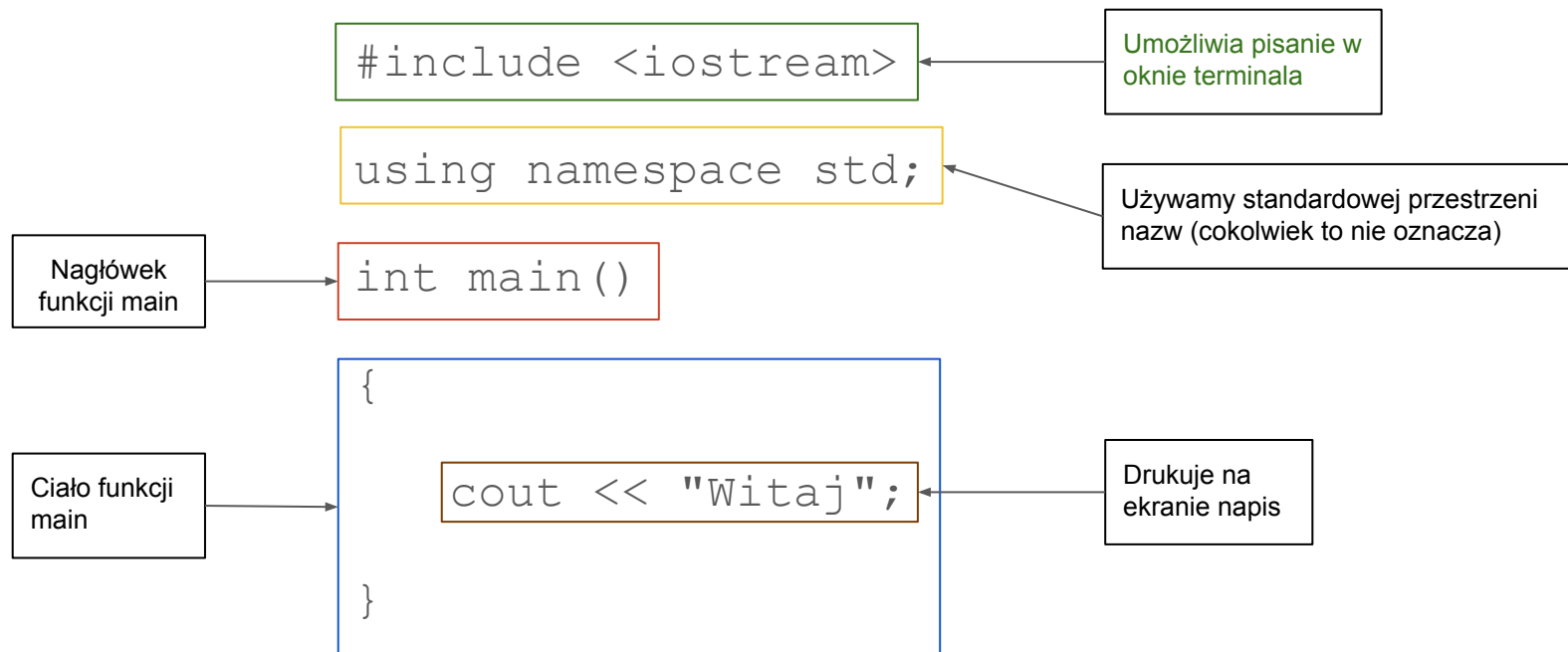
# Programowanie komputerowe

Zajęcia 1

# Code::Blocks - tworzenie projektu

- Create New Project
- Console Application -> C++
- Wybierz nazwę projektu
- Stworzy się nowy projekt z wpisaną funkcją main
- Wpisz swój program
- Skompiluj i uruchom za pomocą F9

# Pierwszy program



# Zmienne

- Zmienne to obiekty pozwalające przechowywać dane.
- Każda zmienna ma swój typ, który mówi jakiego rodzaju wartości można w niej przechować.
- Typy zmiennych to m.in.
  - int - liczby całkowite
  - char - znaki
  - double - liczby rzeczywiste
  - i wiele innych

- Żeby używać zmiennej, trzeba ją najpierw zadeklarować:

```
nazwa_typu nazwa_zmiennej;
```

na przykład

```
int a; // zmienna a przechowująca liczby całkowite
```

```
char litera; // zmienna litera przechowująca znaki
```

# Stałe

Stałe wyrażają pewne wartości ustalone w trakcie pisania programu.

- Stałe całkowite (`int`), np.

4            -11            2878

- Stałe rzeczywiste (`double`), np.

1.5            12.0            -36.1232            2.17e21

- Stałe znakowe (`char`), np.

'A'            '5'            '\n'

- Stałe napisowe (`string`), np.

"ABCD"            "15"            "A"            ""

Uwaga: Apostrofów i cudzysłowów nie można używać zamiennie!

# Przypisanie

Przypisanie służy do ustawienia wartości zmiennej.

Składnia: `nazwa_zmiennej = wartość;`

Przykłady:

- `a=4;` // ustawia wartość zmiennej a na 4
- `a=a+1;` // zwiększa wartość zmiennej a o 1
- `a=b+c;` // ustawia wartość a na sumę wartości zmiennych b i c
- `int a=3;` // ustawienie wartości w momencie deklaracji

# Pisanie na ekranie

W C++ do pisania na ekranie terminala służy obiekt `cout` zadeklarowany w bibliotece `iostream`.

Składnia:

```
cout << wyrażenie1 << wyrażenie2 << ... << wyrażenien;
```

Przykład:

- `cout << "a+b" << '=' << a+b << endl;`

`endl` jest specjalnym znakiem powodującym przejście do następnej linii.

# Wczytywanie danych z klawiatury

Do wczytywania danych służy obiekt `cin` (biblioteka `iostream`). Przykładowy program pokazujący jak wczytać dane:

```
int main() {  
    int n;  
    cout << "Podaj liczbę: ";  
    cin >> n;  
    cout << "Podałeś liczbę " << n << "." << endl;  
}
```



# Ćwiczenie

Napisać program, który pyta użytkownika o dwie liczby (całkowite), a następnie drukuje na ekranie ich sumę i iloczyn.

```
Podaj pierwszą liczbę: 8  
Podaj drugą liczbę: 13  
8+13=21  
8*13=104
```

# Komentarze

Komentarz to część programu, która nie jest kompilowana. W C++ są dwa rodzaje komentarzy:

- `//` zaczyna komentarz do końca linii
- `/*` zaczyna komentarz, który kończy się przez `*/`

W komentarzu umieszczamy wyjaśnienia dotyczące działania programu. Można również wykomentować instrukcje, których chwilowo nie chcemy wykonywać.

# Operatory

Operatory służą do budowania wyrażeń. Oto ich (niekompletna) lista:

Operatory dla liczb:

- + dodawanie
- - odejmowanie
- \* mnożenie
- / dzielenie - całkowite dla liczb całkowitych, zwykle dla rzeczywistych
- % reszta z dzielenia (tylko liczby całkowite)
- potęgowania nie ma

# Operatory (2)

## Operatory porównania:

- == równe
- != różne
- < mniejsze
- <= mniejsze lub równe
- > większe
- >= większe lub równe

## Operatory logiczne:

- ! negacja (nie)
- && koniunkcja (i)
- || alternatywa (lub)

## Operatory zwiększania i zmniejszania:

- ++
- --

# Instrukcje sterujące

Instrukcje w funkcji wykonywane są kolejno - od pierwszej do ostatniej. Można to zmienić przy użyciu instrukcji sterujących. Instrukcje sterujące dzielą się na:

- instrukcje warunkowe (`if`, `switch`) - uzależniające wykonanie grupy instrukcji od pewnego warunku,
- instrukcje pętli (`for`, `while`, `do`) - pozwalające na wykonanie pewnych instrukcji wiele razy,
- wywołania funkcji.

# Instrukcja warunkowa `if`

Wersja 1:

```
if (warunek) {  
    lista_instrukcji  
}
```

Jeśli spełniony jest warunek,  
wykonujemy instrukcje w klamerkach;  
jeśli nie, nie robimy nic.

Wersja 2:

```
if (warunek) {  
    lista_instrukcji_1  
}  
else {  
    lista_instrukcji_2  
}
```

W zależności od warunku  
wykonujemy listę 1 lub 2.

# Instrukcja if - przykład

```
int main() {
    int a,b;
    cout << "Podaj pierwszą liczbę: ";
    cin >> a;
    cout << "Podaj drugą liczbę: ";
    cin >> b;
    if (a>b)
        cout << "Pierwsza jest większa.";
    else
        if (a<b)
            cout << "Druga jest większa.";
        else
            cout << "Obie są równe.";
}
```

Jeśli po `if` (lub `else`) chcemy wykonać tylko jedną instrukcję, klamerki można pominąć.

# Instrukcja pętli `while`

Składnia:

```
while (warunek) {  
    lista_instrukcji  
}
```

Jeśli lista instrukcji jest jednoelementowa, klamerki można pominąć.





# Instrukcja `while` - przykłady

```
int main() {  
    int i=1;  
    while (i<=10) {  
        cout << i << ", ";  
        i=i+1;  
    }  
}
```

Drukuje liczby 1...10.

```
int main() {  
    int i=1;  
    int j=3;  
    while (i<=1000) {  
        cout << i << ", ";  
        i=i+j;  
        j=j+2;  
    }  
}
```

To też drukuje pewne liczby - jakie?

## Instrukcja `while` - przykłady (2)

```
int main() {
    int i=1;
    while (i>0) {
        cout << i << ", ";
        i=i+1;
    }
}
```

Ten program się nie skończy.

```
int main() {
    int i;
    cin >> i;
    while (i!=1) {
        cout << i << ", ";
        if(i%2==0) i=i/2;
        else i=3*i+1;
    }
}
```

Czy ten program zawsze się kończy?

# Zadania

1. Napisać program, który wypisuje wszystkie liczby od 100 do 1 (malejąco).
2. Napisać program, który wypisuje wszystkie liczby od 100 do 1 (malejąco), które nie są podzielne przez 3.
3. Napisać program, który wypisuje potęgi liczby 2 nie większe niż  $n$ . Liczba  $n$  jest podana przez użytkownika.
4. Napisać program, który pyta użytkownika o liczby aż zostanie wpisane 0. Wtedy program wypisuje sumę podanych liczb.
5. Napisać program, który drukuje tabliczkę mnożenia.