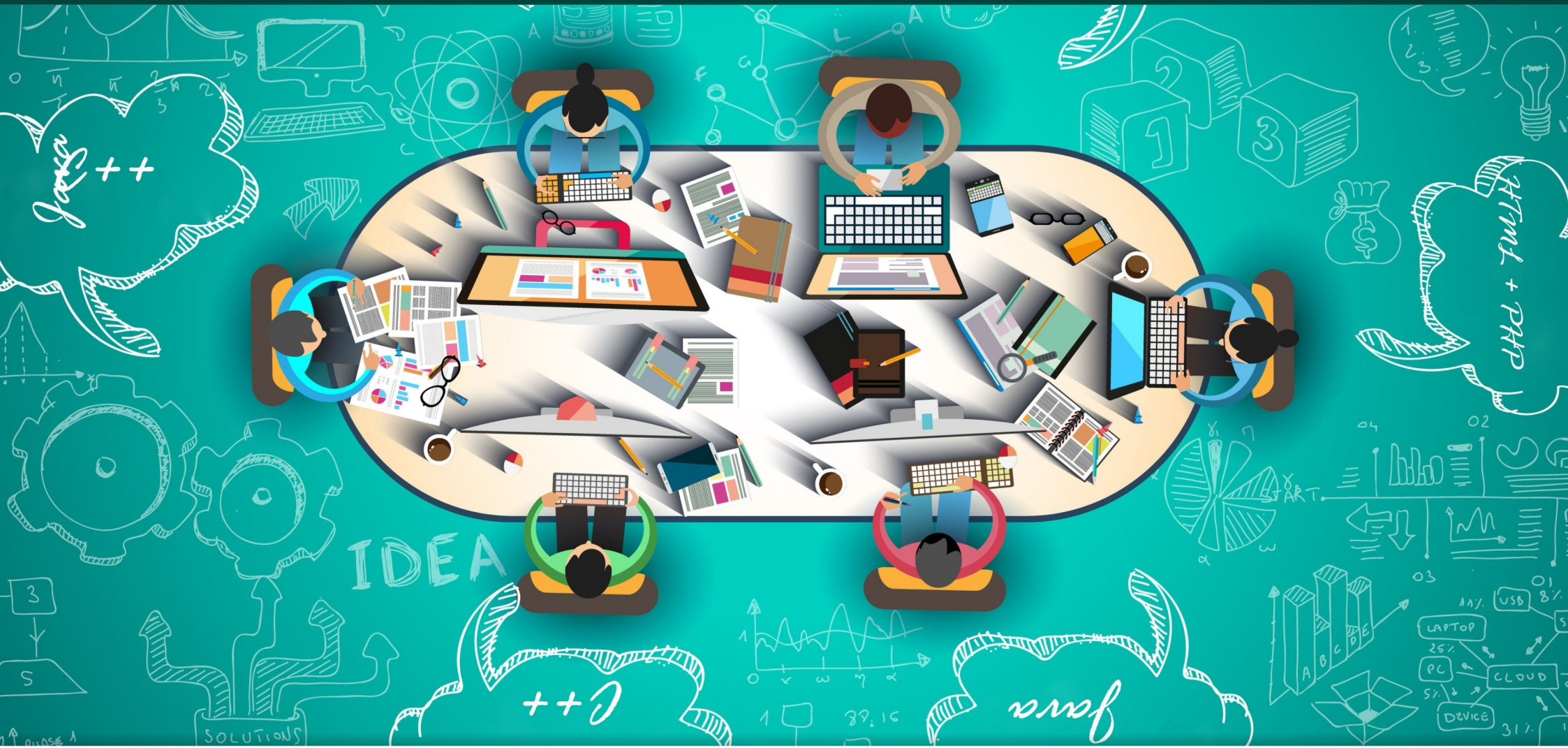
Young Programmer

www.inceptio.org.pl



Organizator:

fundacja INCEPTIO

Partnerzy:

Wydział Fizyki Politechnika Warszawska

Stowarzyszenie KLATRAT

Dofinansowanie:

MIASTO STOŁECZNE WARSZAWA

Zakochaj się w Warszawie

Young Programmer

www.inceptio.org.pl

Zajęcia nr 1

Podstawy programowania

dr inż. Łukasz Graczykowski
mgr inż. Leszek Kosarzewski
Wydział Fizyki Politechniki Warszawskiej

Organizator:

Partnerzy:

Dofinansowanie:

MIASTO STOŁECZNE WARSZAWA

Ramowy program warsztatów

1. **Pierwsze:** Podstawy programowania
2. **Drugie:** Programowanie strukturalne
3. **Trzecie:** Programowanie obiektowe
4. **Czwarte:** Kontenery i algorytmy STL
5. **Piąte:** Biblioteki graficzne i grafika 2D

Materiały do warsztatów:
http://www.if.pw.edu.pl/~lgraczyk/wiki/index.php/Young_Programmer
- zadania, polecana literatura, najważniejsze informacje
http://yp.karpiarz.net/ - instrukcje wspólne

Na czym pracujemy?

System operacyjny: Linux Debian 8.2

Kompilator: GCC 4.9.2

Edytor: KATE

Komendy systemu linux

ls (list) - wyświetla zawartość bieżącego katalogu lub katalogu podanego jako parametr.
cd (change directory) - wchodzi do katalogu, no. cd katalog1
mkdir (make directory) - do tworzenia katalogów. Przykład: mkdir nazwa_katalogu
cp (copy) - do kopiowania plików i katalogów. Przykłady: cp plik1 plik2
cp -r - kopiuje katalog wraz z zawartością np. cp -r katalog1 katalog2
* - gwiazdka zastępuje dowolny ciąg znaków np.:
cp * alfa/ - kopiuje wszystkie pliki z bieżącego katalogu do katalogu alfa
mv (move) - przenosi plik/pliki, służy też do zmiany nazwy pliku lub katalogu.
mv plik1 plik2 - zmienia nazwę plik1 na plik2
rm (remove) - usuwa pliki. Przykład:
rm plik1 - usuwa plik1
rm * - usuwa wszystkie pliki z bieżącego katalogu (należy używać bardzo ostrożnie - sprawdzić, czy rzeczywiście chcemy wszystko skasować).
rm -r - usuwa cały katalog razem z zawartością
more - pozwala na przeglądanie danych (plików, komunikatów poleceń) ekran po ekranie.
kate plik.txt – uruchamia edytor kate tworząc plik plik.txt
cat - podobnie do polecenia 'more' pokazuje zawartość pliku ale nie zatrzymuje się ekran po ekranie tylko wyświetla od razu całość.

Pierwszy program

```
/*****************
 * Jan Kowalski   *
 * 15.03.2013 r.  *
 *****************/



#include <iostream>



int main ()
{

        // Wyświetla linijkę tekstu
        std::cout << "Moj pierwszy
        Program!" << std::endl;

        return 0;  // kończy program
}
```

Komentarz blokowy – dowolny tekst pomiędzy znakami /* oraz */

Instrukcja preprocesora – zaczyna się od znaku **#**

Funkcja main() – tutaj zaczyna się sterowanie programem

Instrukcja – linijki na ogół kończą się **średnikiem**

Komentarz – zaczyna się od // kończy wraz z końcem linii

Prosta kompilacja programu – Linux

Plik z kodem źródłowym: program00.cpp
(pliki z kodem źródłowym języka C++ powinny mieć rozszerzenie .cpp)
Plik wynikowy: progam00
(w środowisku linux programy nie posiadają rozszerzenia, lecz wyróżnia je flaga wykonywalności '*x*')

```
g++ -o program00 program00.cpp -Wall -pedantic
```

Flagi kompilacji:
-Wall – wyświetla wszystkie ostrzeżenia
-pedantic – wyświetla niezgodności ze standardem ISO

Wypisywanie na ekran

```
int main ()
{
    std::cout << "Hello world!" << std::endl;
    return 0;
}
```

cout << " Napis"; // pisanie po ekranie

Wczytaj i wypisz

```
int main ()
{
    int n;
    std::cin >> n;
    std::cout << n << std::endl;
    return 0;
}
```

cin >> zmienna; // standardowe wejście (klawiatura) wpisz do zmiennej

cout << zmienna; // na standardowe wyjście (ekran) wypisz zmienną

Wczytaj i wypisz

```
int main ()
{
    int n;
    std::cin >> n;
    std::cout << n << std::endl;
    return 0;
}
```

cin >> zmienna; // standardowe wejście (klawiatura) wpisz do zmiennej

cout << zmienna; // na standardowe wyjście (ekran) wypisz zmienną

Pętla "for"

```
int main ()
{

        int n;
        std::cin >> n;
        std::cout << n << std::endl;

        for (int i=1;i<=n;i++)
        {
                std::cout<<i; // ...
        }

        return 0;
}
```

Pętla "for": (int i=1;i<=n;i++)
Zaczynając od i równego 0 (int i = 1), do i mniejszego równego n (i<=n), wykonuj raz po raz to co jest w pętli { ... }, przy każdej iteracji zwiększając i (i++)

Czyli: n razy wykonaj to, co jest w pętli za każdym razem zwiększając i

Instrukcja warunkowa "if"

```
int main ()
{
        int n;
        std::cin >> n;
        if (n >= 0)
        {
                long silnia = 1;
                for (int i = 1; i <= n; i++)
                        silnia *= i;
                std::cout << "Silnia wynosi " << silnia << std::endl;
        }

        return 0;
}
```

Jeśli n większe równe 0

wtedy rób to co w klamrach

Instrukcja warunkowa "if" oraz "else"

```
int main ()
{
        int n;
        std::cin >> n;
        if (n >= 0)
        {
                long silnia = 1;
                for (int i = 1; i <= n; i++)
                        silnia *= i;
                std::cout << "Silnia wynosi " << silnia << std::endl;
        }
        else
        {
                std::cout << "Nie moge obliczyc silni z liczby ujemnej!\n";
        }
        return 0;
}
```

Jeśli n większe równe 0

wtedy rób to co w klamrach

W przeciwnym wypadku

wtedy rób to co w kolejnych klamrach

Indentacja

```
int main ()
{
        int n;
        std::cin >> n;
        if (n >= 0)
        {
                long silnia = 1;
                for (int i = 1; i <= n; i++)
                        silnia *= i;
                std::cout << "Silnia wynosi " << silnia << std::endl;
        }
        else
        {
                std::cout << "Nie moge obliczyc silni z liczby ujemnej!\n";
        }
        return 0;
}
```

Brak wcięć nie powoduje błędów kompilacji, jednak prawidłowe używanie wcięć zwiększa czytelność kodu!

Indentacja – *Python*

```
def add5(x):
    return x+5

def dotwrite(ast):
    nodename = getNodename()
    label=symbol.sym_name.get(int(ast[0]),ast[0])
    print '    %s [label="%s' % (nodename,label),
    if isinstance(ast[1], str):
        if ast[1].strip():
            print '= %s"];' % ast[1]
        else:
            print '"]'
    else:
        print '"];'
        children = []
        for n, child in enumerate(ast[1:]):
            children.append(dotwrite(child))
        print '    %s -> {' % nodename,
        for name in children:
            print '%s' % name,
```

W ostatnio popularnym języku *Python* wcięcia stanowią ważny element języka – wskazują na bloki kodu (zamiast klamer stosowanych w C/C++)

Young Programmer

www.inceptio.org.pl

Zajęcia nr 1

Podstawy programowania

Zadania

Organizator:

fundacja INCEPTIO

Partnerzy:

Wydział Fizyki Politechnika Warszawska

Stowarzyszenie KLATRAT

Dofinansowanie:

MIASTO STOŁECZNE WARSZAWA

Zakochaj się w Warszawie

(1) "Hello world"

Wypisać na ekran (w terminalu) słowa "Hello World!"

- tworzymy nowy plik tekstowy, nadajemy mu nazwę hello.cpp
- na początku załączamy bibliotekę: **#include <iostream>**
- określamy przestrzeń nazw: **using namespace std;**
- tworzymy funkcję main

```
int main()
{

 return 0;
}
```

- w środku funkcji wypisujemy słowo przy użyciu "cout" : cout<<"Napis!"<<endl;
- kompilujemy – w terminalu wpisujemy:

```
g++ -Wall hello.cpp -o hello
```

(2) "Zmienne, cout, cin"

Stworzyć funkcję główną (main) w której należy kolejno:

- zadeklarować zmienną całkowitą a = 5 i wypisać ją na ekran (int a = 5;)

- zadeklarować zmienną zmiennoprzecinkową b = 3.5 i wypisać ją na ekran (double .... ;)

- zadeklarować zmienną zmiennoprzecinkową c która będzie wynikiem sumowania zmiennych a i b (c = a + b;)

- wypisać na ekran napis: a + b = c oraz odpowiednio to samo równanie używając wartości zmiennych (wskazówka: cout<<"napis"<<a<<" + "<<b<<endl;)

- zadeklarować zmienną typu "string" (napis) "Oto jestem napisem". Wypisać ją na ekran.

- wypisać na ekran napis "Ile masz lat?"

- poprosić użytkownika programu o wprowadzenie liczby z klawiatury (cin >> d) (wiek jest liczbą całkowitą!)

- wypisać podany przez użytkownika wiek w postaci "Mam X lat"

- poprosić użytkownika o wprowadzenie imienia z klawiatury (cin >> imie) (imię jest napisem!)

- wypisać na ekranie "Nazywam się XXX"

(3) "Jeśli"

Stworzyć funkcję główną (main) w której należy kolejno:

- wypisać na ekranie "Ile masz lat?"

- poprosić użytkownika o wprowadzenie liczby całkowitej z klawiatury

- jeśli użytkownik podał wiek mniejszy niż 18 lat wypisać: "Nie masz 18 lat!" , jeśli większy to wypisać " Masz XXX lat i możesz przeczytać ten tekst!"

Przykład użycia w kodzie programu "jeśli"

```
if(a > 5)
  {
    cout<<"Liczba a jest większa niż 5!"<<endl;
  }
else
 {
   cout<<"Liczba a jest mniejsza niż 5!"<<endl;
 }
```

(3.5) "Jeśli"

Stworzyć funkcję główną (main) w której należy kolejno:

- wypisać na ekranie "Ile masz lat?"

- poprosić użytkownika o wprowadzenie liczby całkowitej z klawiatury

- jeśli użytkownik podał wiek 3 - 18 lat wypisać: "Nie masz 18 lat!" , jeśli większy to wypisać " Masz XXX lat i możesz przeczytać ten tekst!"

- jeśli użytkownik podał wiek poniżej 3 lat wypisać "Kłamiesz!"

Przykład użycia w kodzie programu "jeśli"

```
if(a > 5)
 {
   cout<<"Liczba a jest większa niż 5!"<<endl;
 }
else
 {
  cout<<"Liczba a jest mniejsza niż 5!"<<endl;
 }
```

Znak **&&** oznacza "**i**"
Znak **||** oznacza "**lub**"

np. if (i < 0 || i >100)
cout<<"Zly wiek";

//Jeśli i jest mniejsze od zera bądź większe od 100 to wypisz "Zły wiek"

(4) Naucz brata dodawania

Stworzyć funkcję główną (main) w której należy kolejno:
- stworzyć pętlę while

```
bool koniec = true;
while(koniec)
{
...
}
```

A wewnątrz pętli:
- pobrać od użytkownika pojedynczy znak z klawiatury (char a, cin >> a)
- w zależności od podanego znaku wykonać jedną z trzech rzeczy (1), (2) lub (inne)

Do tego celu służy funkcja switch-case

```
switch(a){
   case '1':
     ....
   break;
   case '2':
     ....
   break;
   default':
     ....
   break;
}
```

- jeśli "default" (**wartość domyślna = inna niż wymienione**) to wypisz na ekran "Podaj liczbę 1 lub 2!"
- jeśli (2) to wyjdź z programu (zmienną **koniec** należy ustawić na **false**)
- jeśli (1) to poproś użytkownika o podanie dwóch liczb, następnie poproś użytkownika o podanie sumy tych dwóch liczb. Jeśli podał prawidłową wartość, wypisz "Poprawny wynik!" jeśli zaś nieprawidłowy, to wypisz "Wynik niepoprawny, poprawny wynik to XXX".

Więcej o switch-case:
http://cpp0x.pl/kursy/Kurs-C++/Poziom-1/Warunek-wielokrotnego-wyboru-switch-case/17

(5) Kalkulator

Korzystając z opanowanych umiejętności należy stworzyć program KALKULATOR. Powinien posiadać następujące funkcje:

Dodawanie
Odejmowanie
Mnożenie
Dzielenie
Pierwiastek (sqrt)
Potęgowanie (pow)
Wartość absolutna (abs)
1/x
Silnia

```
#include <math.h>
```

Przykład użycia funkcji:

```
pierwiastek_z_2 = sqrt(2);
wartosc_bezwzgl = abs(-1);
```