

## Ogólne zalecenia:

- Programy z danego tygodnia powinny mieć własne katalogi.
- Każdy program ma mieć własny plik/pliki źródłowe i własny plik wykonywalny.
- Kompilujemy z nadaniem nazwy i flagami -Wall oraz -std=c++11 :  
**g++ -Wall -std=c++11 -o nazwa\_zrodlo.cpp**
- Zawsze myślimy o przypadkach szczególnych (np. w zadaniu z f. kwadratową możliwe było podanie  $a=0$ ).
- Programy mają być idioto-odporne, tzn. zawsze zakładamy, że użytkownik użyje go w zły sposób (chyba, że w zadaniu jest napisane dla prostoty, że nie musimy czegoś sprawdzać). Oznacza to, że musimy sprawdzać, czy dane z konsoli czy zewnętrznego pliku mają sensowne wartości itp.
- Niedopuszczalne jest, żeby program po niepoprawnym użyciu nie wyświetlił żadnej informacji zwrotnej.
- Programy muszą komunikować się z użytkownikiem, np. prosić o podanie liczby itp.
- Wypisywane na konsolę wyniki działania programu mają być czytelne, należy stosować białe znaki ' ' oraz znaki nowej linii '\n' lub std::endl.
- Należy **ściśle** trzymać się wytycznych. Jeżeli program ma coś robić po napisaniu 's' to ma to robić po napisaniu 's' a nie np. 'a'.
- Kod powinien być czytelny, nazwy zmiennych powinny mieć sens.
- Należy robić wcięcia.
- To co nie jest zabronione, jest dozwolone.

## ZADANIE 0

Napisz program z dwiema funkcjami rekurencyjnymi:

```
int silnia(int n)  
double potega(double x, int k)
```

Pierwsza funkcja liczy silnię z liczby naturalnej. Druga funkcja podnosi liczbę rzeczywistą do potęgi naturalnej.

**Pamiętaj o warunku na zakończenie rekurencji.**

Program ma poprosić użytkownika o podanie  $n, x, k$  oraz wyświetlić czytelnie wyniki. Ma to wyglądać mniej więcej tak:

```
Podaj liczbę naturalną n  
3  
Podaj liczbę rzeczywistą x  
1.5  
Podaj potęgę naturalną k  
4
```

```
3! = 6
1.5^4 = 5.0625
```

Można założyć, że są  $n$  i  $k$  są całkowite.

### ZADANIE 1

Napisać program, który prosi użytkownika o wpisanie liczby w systemie binarnym i wyświetla liczbę w systemie dziesiętnym. Można założyć, że liczba jest nie większa niż 1024. Zakładamy, że użytkownik podaje liczbę dodatnią zapisaną poprawnie w systemie dwójkowym, bez 0 z przodu.

```
Podaj liczbę naturalną w systemie binarnym
1111011110
Liczba w systemie dziesiętnym:
990
```

### ZADANIE 2

Wykorzystaj (skopiuj) funkcje z zadania 0 i napisz program, który liczy przybliżoną wartość funkcji  $\sin(x)$  korzystając z rozwinięcia w szereg McLaurina. Napisz dwie nowe funkcje, jedną liczącą wartość  $n$ -tego wyrazu, drugą sumującą wyrazy od zerowego aż do  $n$ -tego:

```
long double wyraz(long int n, long double x)
long double ciag(long int n, long double x)
```

W funkcji *ciag* użyj rekurencji. Działanie programu wygląda następująco:

```
Podaj nr wyrazu n
8
Podaj argument x
3.14159
Sinus(3.14159) = 2.67601e-06 z dokładnością do 8 wyrazu.
```

**UWAGA:** Zadbaj o jak najlepszą dokładność obliczeń.

### ZADANIE 3

Napisać grę w 20 pytań. Użytkownik myśli bądź zapisuje na kartce pewną liczbę z przedziału  $[1-10^6]$ . Program zadaje w pętli pytania: *Czy to n?* na co użytkownik odpowiada:

- s (jak small) jeśli jego liczba jest mniejsza niż wyświetlana
- b (jak big) jeśli jego liczba jest większa niż wyświetlana
- y (jak yes) jeśli jego liczba jest wyświetlana

Program powinien kończyć się wyświetleniem komunikatu

**Pomyślana liczba to...**

i podaniem prawidłowej odpowiedzi oraz liczby pytań.

**UWAGA:** Nie używać tablic, napisów ani żadnych innych kolekcji.