Ćwiczenia 4. Instrukcje iteracyjne

Zadanie 1.

Zmodyfikować funkcję `oblicz_silnie()` (program pp_cw04_1_silnia.cpp) tak, aby zwracała:

- wartość $n!$,jeżeli $0 \leq n \leq 20$
- wartość -1, jeżeli $n < 0$;
- wartość -2, jeżeli $n > 20$;

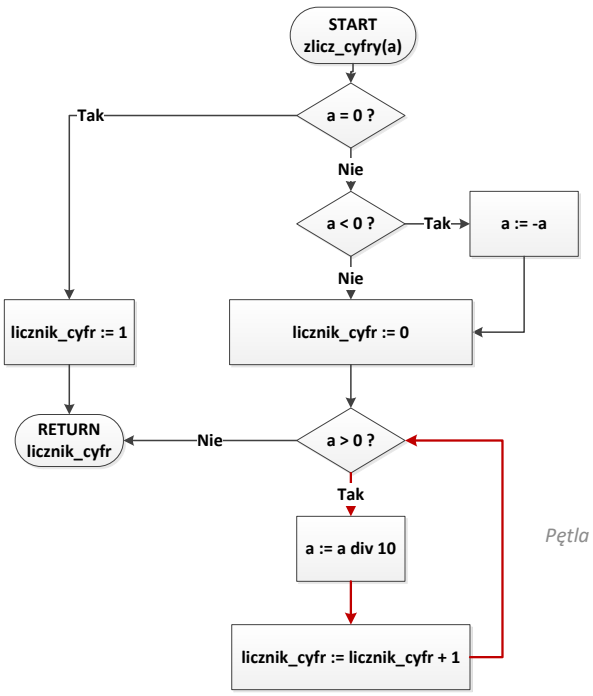
A więc funkcja `oblicz_silnie()` ma być sama odpowiedzialna za swoje działania w zależności od przekazanej do niej wartości parametru n .

Wywołać funkcję `oblicz_silnie()` w funkcji `main()` wykorzystując wartość przez nią zwracaną w następujący sposób:

```
s = oblicz_silnie(n);
if(s == -1)  //jeżeli nie wykonano obliczeń z powodu ujemnej wartosci n
      //TODO: wyświetl komunikat o ujemnej wartości n
else if(s == -2) //jeżeli nie wykonano obliczeń z powodu za dużej wartosci n
      //TODO: wyświetl komunikat o za dużej wartości n
else       //w pozostałych przypadkach (tzn. jeżeli obliczenia zostały wykonane)
      //TODO: wyświetl wartość n!
```

Zadanie 2.

Rozszerzyć funkcję `zlicz_cyfry(a)` (przykład 3 wykładu PP05, program pp_cw04_2_zliczanie_cyfr.cpp) wyznaczającą ilość cyfr liczby całkowitej `a` tak, aby działała ona według poniższego schematu blokowego.



Uwagi:

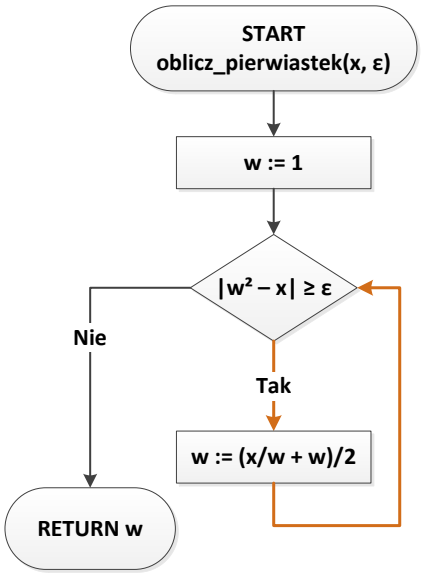
- operacja **div** na schemacie blokowym oznacza dzielenie całkowite;
- operacja **:=** oznacza przypisanie;
- liczba o wartości 0 składa się z jednej cyfry;
- liczba ujemna, np. -7624 składa się z czterech cyfr (zliczamy tylko cyfry, zapominamy o znaku).

Zadanie 3.

Napisać funkcję `oblicz_pierwiastek(x,ε)` obliczającą i zwracającą wartość pierwiastka kwadratowego liczby `x` według algorytmu (metoda babilońska) przedstawionego na poniższym schemacie blokowym. Zmienna `ε` jest parametrem reprezentującym dokładność wykonanych obliczeń, zmienna `w` przechowuje wynik obliczania pierwiastka z `x`.

Sprawdzić, jakie wartości `w` zostaną otrzymane przy różnych wartościach parametru `ε`: `ε` = 0.1, 0.01, 0.001, 0.0001.
Ile iteracji (obrotów pętli) wykona się przy `ε` = 0.1, 0.01, 0.001 i 0.0001.

Wyświetlić wynik z dokładnością do **8** miejsc po przecinku.
Należy użyć instrukcję iteracyjną **`while`**.



Zadanie 4.

Napisać i przetestować funkcję obliczającą i zwracającą wartość x^p, określoną następująco:

$$x^0 = 1$$
$$x^p = x \cdot x \cdot x \cdot \ldots \cdot x \quad \text{dla } p > 0$$
$$x^p = \frac{1}{x^{-p}} \quad \text{dla } p < 0$$

gdzie x jest liczbą rzeczywistą, a p jest liczbą całkowitą (ujemną lub dodatnią).

Wyświetlić wynik z dokładnością do 3 miejsc po przecinku.
Należy zastosować instrukcję iteracyjną **`for`** (NIE należy korzystać z funkcji bibliotecznych, np. pow() – w zadaniu tworzymy swoją własną funkcję „jakby biblioteczną”).
Narysować schemat blokowy implementowanego algorytmu.

Zadanie 5.

Napisać program znajdujący wszystkie liczby pierwsze w zakresie podanym przez użytkownika (np. w zakresie od 21 do 30 są dwie liczby pierwsze: 23 i 29). Skorzystać z funkcji `czy_pierwsza()` z programu pp05_04_czy_pierwsza.cpp.

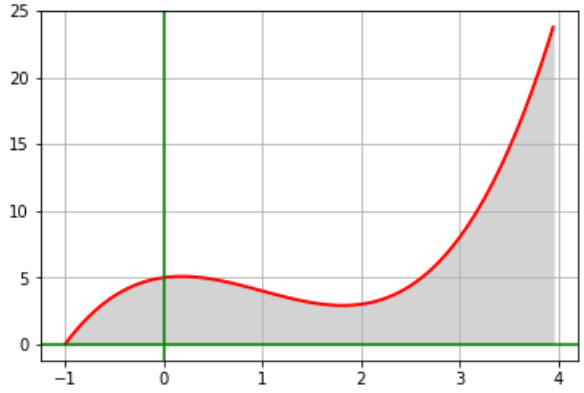
Zadanie 6* (+0.5 punkta**).**

Napisać program obliczający pola powierzchni figur zaznaczonych na szaro na poniższych rysunkach:

a) figury ograniczonej przez wykres krzywej $f(x) = x^3 - 3x^2 + x + 5$, oś x i proste $x = -1,\ x = 4$
b) figury ograniczonej przez wykres krzywej $f(x) = x^3$, oś x i proste $x = -10, x = 15$

Można skorzystać z funkcji `oblicz_calke()` z programu pp05_05_calka.cpp.

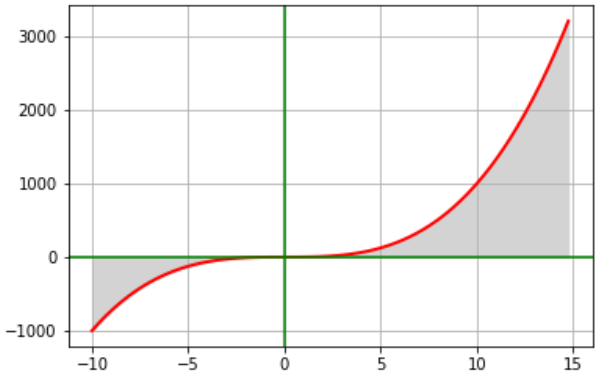
a)



$f(x) = x^3 - 3x^2 + x + 5$

Odp. pole powierzchni = 31.25

b)



$f(x) = x^3$,
miejsce zerowe: $x = 0$

Odp. pole powierzchni = 15156.25