Faktoryzacja w języku C++

- Wprowadzenie
- Przeczytaj
- Film samouczek
- Sprawdź się
- Dla nauczyciela

Źródło: Franck V., domena publiczna.

Rozkładanie liczby na czynniki pierwsze jest istotnym elementem kryptografii. Własności tego procesu określają, jak trudny do złamania będzie używany przez nas szyfr. Faktoryzacja małej liczby nie stanowi problemu, natomiast rozkład dużych liczb na czynniki pierwsze jest bardzo wymagający obliczeniowo.

W e-materiale Faktoryzacja poznaliśmy algorytm rozkładu liczb na czynniki pierwsze. W tym e-materiale dowiemy się, jak zapisać go w języku C++ oraz jakie ma zastosowanie przy rozwiązywaniu problemów z oprogramowaniem.

Chcesz sprawdzić, jak wyglądają implementacje w innych językach programowania? Możesz się z nimi zapoznać w dwóch pozostałych e-materiałach z tej serii:

- Faktoryzacja w języku Java,
- Faktoryzacja w języku Python.

Więcej zadań? Sięgnij do Faktoryzacja – zadania maturalne.

> **Twoje cele**
>
> - Przeanalizujesz algorytm rozkładu liczby na czynniki pierwsze.
> - Zaimplementujesz poznany algorytm w programie w języku C++.
> - Sprawdzisz swoją wiedzę na temat faktoryzacji w języku C++.

Przeczytaj

Rozkład na czynniki pierwsze

Rozłożenie liczby na czynniki pierwsze polega na zapisaniu jej w postaci iloczynu liczb pierwszych (np. $46 = 2 \cdot 23$). Już od starożytności wiadomo, że każda liczba naturalna większa niż jeden jest albo liczbą pierwszą, albo iloczynem liczb pierwszych (czyli tak zwaną liczbą złożoną).

Metodę rozkładu liczby na czynniki pierwsze poznajemy już w szkole podstawowej, niedługo po opanowaniu tabliczki mnożenia. Jeśli chcesz przypomnieć sobie ten algorytm, zapoznaj się z e-materiałem Faktoryzacja.

Realizacja algorytmu w języku w C++

Napiszemy w języku C++ program rozkładający liczbę naturalną na czynniki pierwsze.

Specyfikacja problemu:

Dane:

- `liczba` – liczba do rozłożenia na czynniki pierwsze; liczba naturalna większa od 1

Wynik:

Program wypisuje wszystkie czynniki pierwsze liczby `liczba` w jednym wierszu, po przecinku.

Zaczniemy od zdefiniowania zmiennej typu `int`. Użyjemy jej do przechowania wartości liczby rozkładanej na czynniki.

```
1 int liczba = 24;
```

Następnie utworzymy pętlę `for`. Będzie ona działać aż do momentu, w którym zmienna `liczba` będzie równa `1`. W pętli deklarujemy zmienną `czynnik`, która będzie oznaczała kolejne rozpatrywane przez nas potencjalne czynniki pierwsze. Będzie ona inkrementowana w każdej iteracji. Jej wartością początkową jest 2, ponieważ jest to najmniejsza liczba pierwsza.

```
1 for (int czynnik = 2; liczba > 1; ++czynnik) {
2
```

```
}
```

Wewnątrz pętli `for` umieścimy pętlę `while`. Pętla `while` wykonuje się, dopóki reszta z dzielenia zmiennej `liczba` przez `czynnik` wynosi 0 (czyli dopóki zmienna `czynnik` jest podzielnikiem rozkładanej liczby).

```
for (int czynnik = 2; liczba > 1; ++czynnik) {
    while (liczba % czynnik == 0) {
        cout << czynnik << ", ";
        liczba /= czynnik;
    }
}
```

Mamy zatem program, który rozłoży daną liczbę na czynniki pierwsze:

```
#include <iostream>

using namespace std;

int main() {
    int liczba = 24;

    for (int czynnik = 2; liczba > 1; ++czynnik) {
        while (liczba % czynnik == 0) {
            cout << czynnik << ", ";
            liczba /= czynnik;
        }
    }
}
```

Możemy jednak stworzyć bardziej optymalny algorytm. Zauważmy, że nie jest konieczne sprawdzanie wszystkich czynników, aż wartość `liczba` wyniesie 1.

Każda liczba złożona, ma maksymalnie jeden czynnik pierwszy większy od jej pierwiastka.

> **Przykład 1**
>
> Liczba 27, po rozkładzie na czynniki pierwsze, to 3·3·3. Żaden z czynników zatem nie jest większy od pierwiastka tej liczby, wynoszącego 5,196...
> **Przykład 2**

> Pierwiastek liczby 14 to 3,74..., a liczba ta po faktoryzacji to 2·7. Widzimy zatem, że tylko jeden czynnik pierwszy jest większy od jej pierwiastka.

W związku z tą obserwacją, możemy przerwać iterację w momencie, gdy `czynnik` podniesiony do kwadratu będzie większy od wartości `liczba`. Na końcu wypiszemy wartość zmiennej `liczba`, jeżeli jest ona większa od 1.

```
#include <iostream>

using namespace std;

int main() {
    int liczba = 24;

    for (int czynnik = 2; czynnik * czynnik <= liczba; ++czynnik)
        while (liczba % czynnik == 0) {
            cout << czynnik << ", ";
            liczba /= czynnik;
        }
    }

    if (liczba != 1) {
        cout << liczba;
    }
}
```

Słownik

> **faktoryzacja**
>
> rozkład liczby na czynniki pierwsze, czyli zapisanie dowolnej liczby naturalnej większej od 1 za pomocą iloczynu liczb pierwszych
> **liczba pierwsza**
>
> liczba naturalna większa od 1, która ma tylko dwa dzielniki, dzieli się tylko przez 1 oraz przez samą siebie
> **liczba złożona**
>
> liczba naturalna większa od 1 będąca iloczynem liczb pierwszych

Film samouczek

Polecenie 1
Napisz program, który pobierze od użytkownika liczbę naturalną, a następnie wskaże rozkład podanej przez użytkownika liczby na czynniki pierwsze.

Specyfikacja problemu:

Dane:

- n – liczba do rozłożenia na czynniki pierwsze; liczba naturalna większa od 1

Wynik:

Program wypisuje, na standardowe wyjście, wszystkie czynniki pierwsze liczby n oddzielone spacją.

Polecenie 2
Porównaj swoje rozwiązanie z zaprezentowanym w filmie.

Trwa wczytywanie danych...

Film dostępny pod adresem /preview/resource/R17ANmE6L093q

Film nawiązujący do rozkładu liczby na czynniki pierwsze.

Plik o rozmiarze 353.00 B w języku polskim

Plik o rozmiarze 476.00 B w języku polskim

Polecenie 3
Uzupełnij definicję funkcji, która rozłoży na czynniki pierwsze twoją datę urodzenia – osobno dzień, miesiąc oraz rok.

Działanie programu przetestuj dla 15.01.1994 r.

Specyfikacja problemu:

Dane:

- `rok` – liczba naturalna dodatnia, należąca do przedziału $[1,\ 2022]$
- `miesiac` – liczba naturalna dodatnia, zawierająca się w przedziale $[1,\ 12]$
- `dzien` – liczba naturalna dodatnia, zawierająca się w przedziale $[1,\ 31]$

Wynik:

Na standardowym wyjściu wyświetlane są czynniki pierwsze liczb `rok`, `miesiac` i `dzien` w osobnych wierszach. Czynniki powinny być oddzielone spacją.

Sprawdź się

Pokaż ćwiczenia: 

Ćwiczenie 1

Zmodyfikuj kod programu w zaznaczonym miejscu tak, aby wypisywał czynniki pierwsze liczby a. Przetestuj jego działanie dla a = 18.

Specyfikacja problemu:

Dane:

- a – liczba do rozłożenia na czynniki pierwsze; liczba naturalna większa od 1

Wynik:

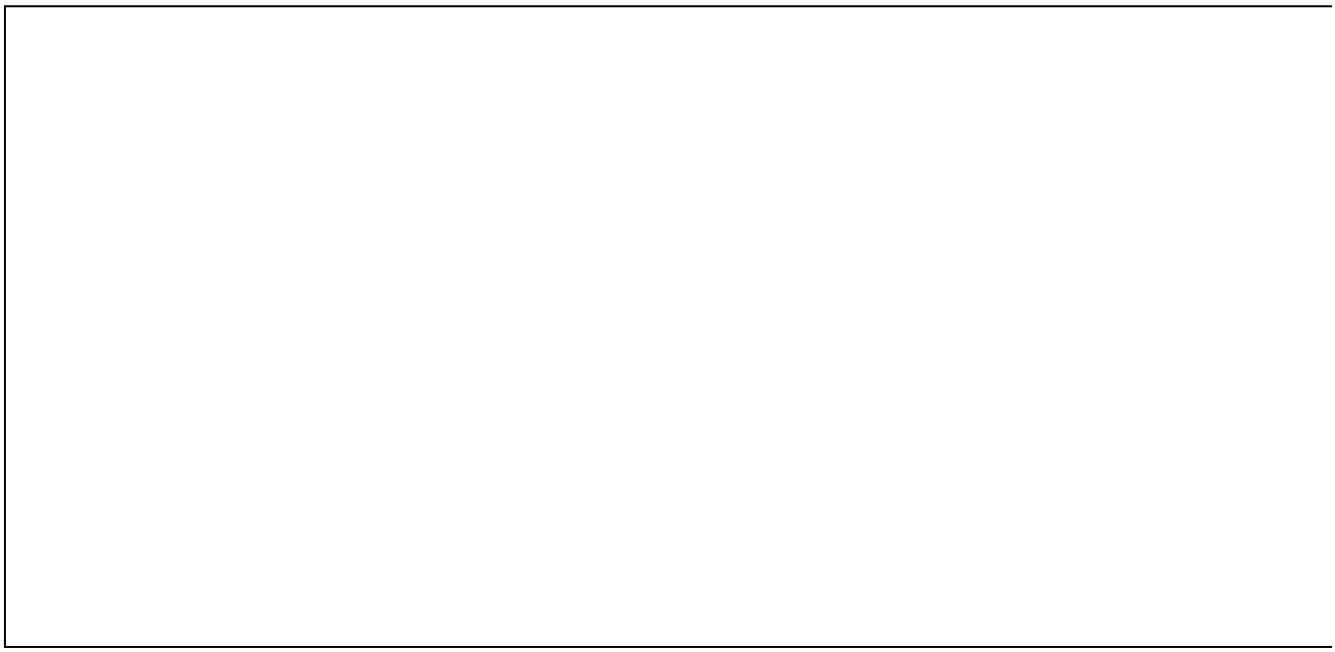
Na standardowym wyjściu program wypisze rozkład liczby a na czynniki pierwsze.

Przykładowe wyjście dla $a = 10$:

```
2 5
```

Twoje zadania

1. Zmodyfikuj kod programu w zaznaczonym miejscu tak, aby wypisywał czynniki pierwsze liczby a.



Ćwiczenie 2

Uzupełnij warunek pętli `for` oraz pętli `while` w programie, który wypisuje czynniki pierwsze liczby naturalnej `a`. Przetestuj działanie programu dla `a = 28`.

Specyfikacja problemu:

Dane:

- `a` – liczba do rozłożenia na czynniki pierwsze; liczba naturalna większa od 1

Wynik:

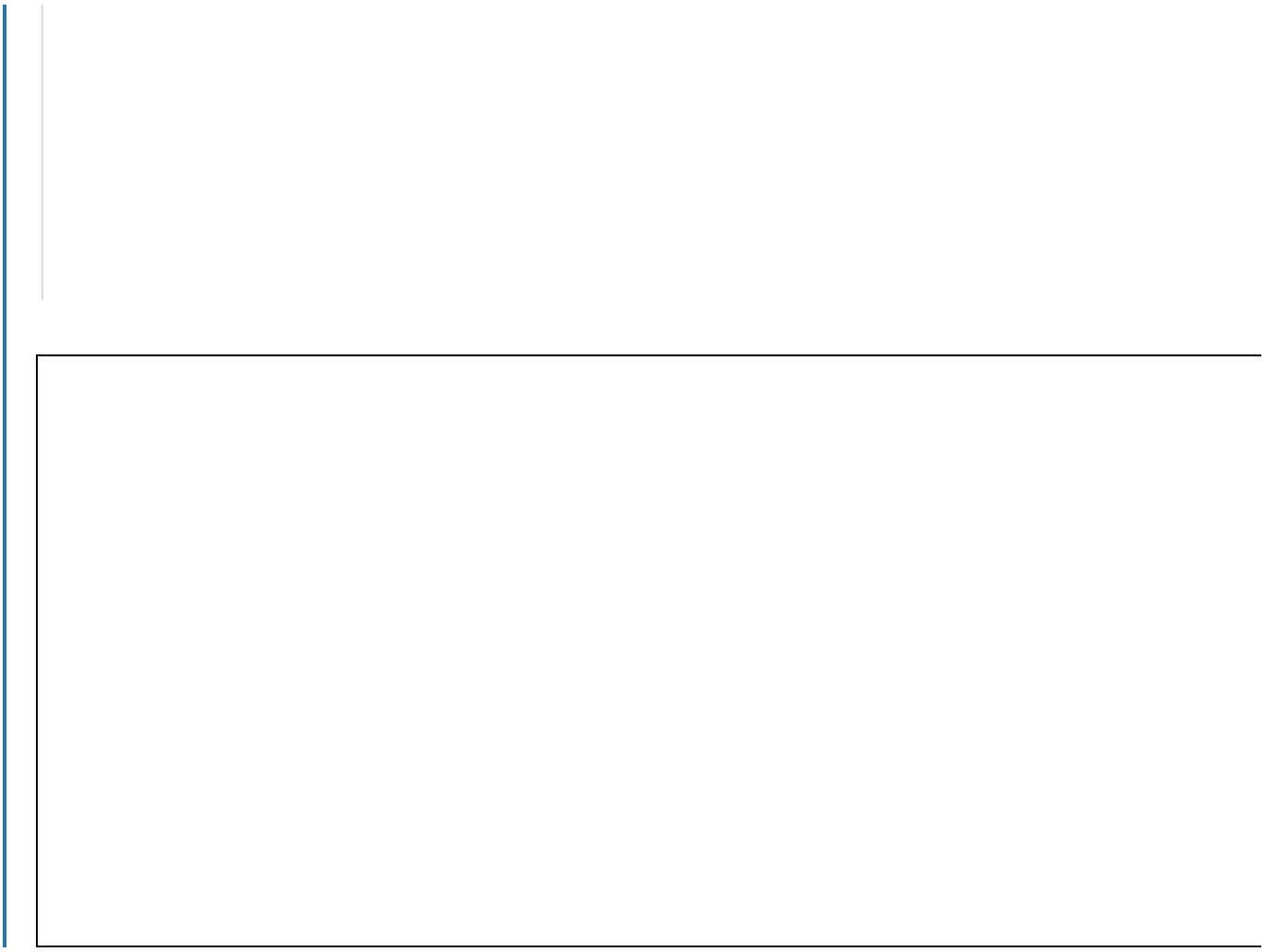
Na standardowym wyjściu program wypisze rozkład liczby `a` na czynniki pierwsze.

Przykładowe wyjście dla $a = 36$:

```
2 2 3 3
```

Twoje zadania

1. Uzupełnij warunek pętli `for` oraz pętli `while` w programie, który wypisuje czynniki pierwsze liczby naturalnej `a`.



Ćwiczenie 3

Napisz program, który oblicza iloczyn największych czynników pierwszych każdej z liczb naturalnych większych od 1. Liczby zostały podane w n-elementowej tablicy a. Przykładowo, dla liczb 17 i 38 ten iloczyn wynosi 323. Przetestuj działanie programu dla tablicy `a = {510, 17, 30, 8, 4000}`.

Specyfikacja problemu:

Dane:

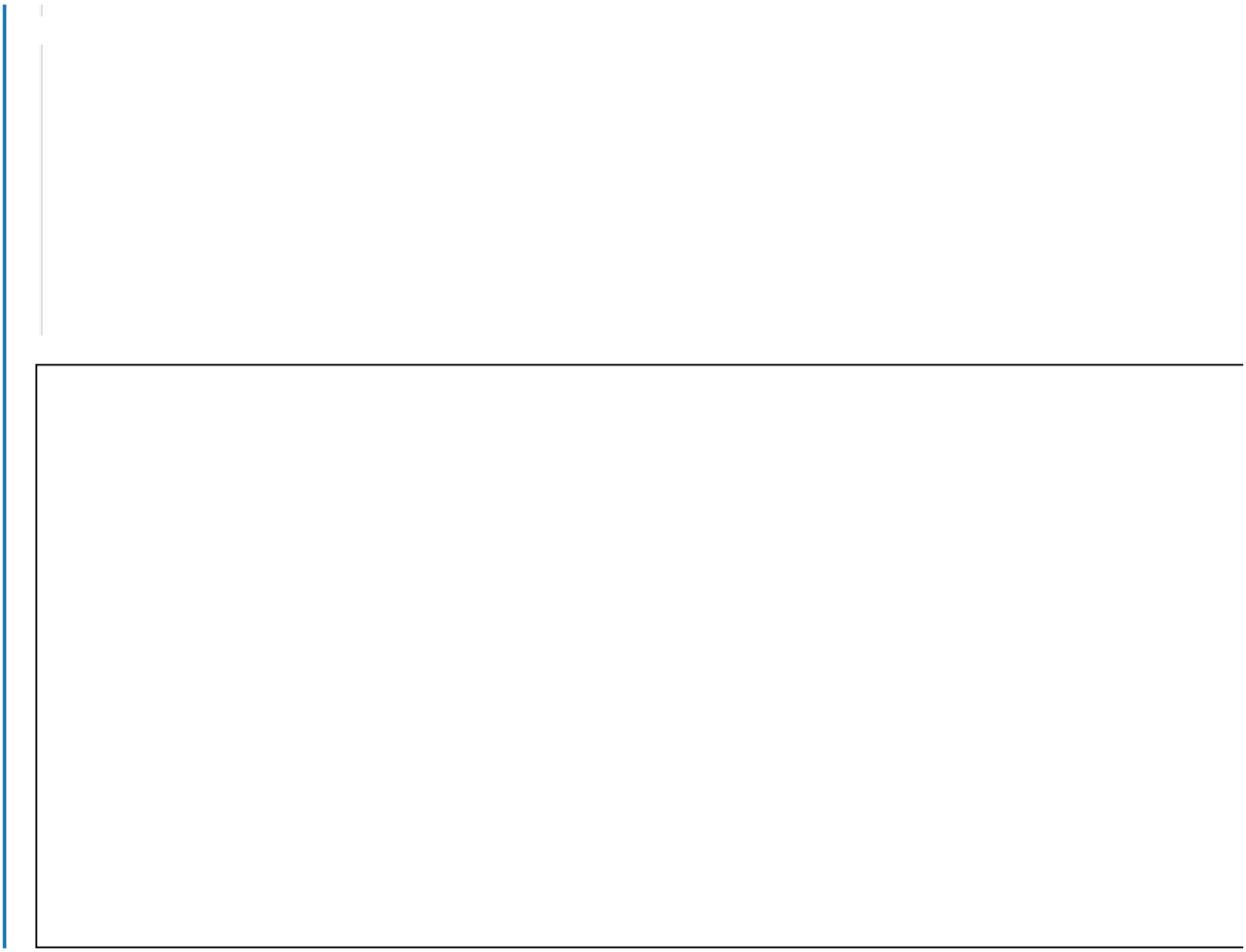
- a – niepusta tablica liczb naturalnych większych od 1
- n – liczba elementów wchodzących w skład tablicy a; liczba naturalna dodatnia

Wynik:

Na standardowym wyjściu wyświetlana jest liczba całkowita: iloczyn największych czynników pierwszych liczb zawartych w tablicy a.

Twoje zadania

1. Program wypisuje jedną liczbę całkowitą, będącą iloczynem największych czynników pierwszych każdej z liczb podanych w tablicy.



Dla nauczyciela

Autor: Maurycy Gast
Przedmiot: Informatyka

Temat: Faktoryzacja w języku C++

Grupa docelowa:

Liceum ogólnokształcące i technikum, liceum ogólnokształcące, technikum, zakres rozszerzony

Podstawa programowa:

Cele kształcenia – wymagania ogólne

I. Rozumienie, analizowanie i rozwiązywanie problemów na bazie logicznego i abstrakcyjnego myślenia, myślenia algorytmicznego i sposobów reprezentowania informacji.

II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera oraz innych urządzeń cyfrowych: układanie i programowanie algorytmów, organizowanie, wyszukiwanie i udostępnianie informacji, posługiwanie się aplikacjami komputerowymi.

Treści nauczania – wymagania szczegółowe

I. Rozumienie, analizowanie i rozwiązywanie problemów.

Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

3) objaśnia dobrany algorytm, uzasadnia poprawność rozwiązania na wybranych przykładach danych i ocenia jego efektywność;

II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera i innych urządzeń cyfrowych.

Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

3) sprawnie posługuje się zintegrowanym środowiskiem programistycznym przy pisaniu, uruchamianiu i testowaniu programów;

I + II. Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

2) wykorzystuje znane sobie algorytmy przy rozwiązywaniu i programowaniu rozwiązań następujących problemów:

a) rozkładania liczby na czynniki pierwsze,

Kształtowane kompetencje kluczowe:

- kompetencje cyfrowe;
- kompetencje osobiste, społeczne i w zakresie umiejętności uczenia się;
- kompetencje matematyczne oraz kompetencje w zakresie nauk przyrodniczych, technologii i inżynierii.

Cele operacyjne (językiem ucznia):

- Przeanalizujesz algorytm rozkładu liczby na czynniki pierwsze.
- Zaimplementujesz poznany algorytm w programie w języku C++.
- Sprawdzisz swoją wiedzę na temat faktoryzacji w języku C++.

Strategie nauczania:

- konstruktywizm;
- konektywizm.

Metody i techniki nauczania:

- dyskusja;
- rozmowa nauczająca z wykorzystaniem multimedium i ćwiczeń interaktywnych;
- ćwiczenia praktyczne.

Formy pracy:

- praca indywidualna;
- praca w parach;
- praca w grupach;
- praca całego zespołu klasowego.

Środki dydaktyczne:

- komputery z głośnikami, słuchawkami i dostępem do internetu;
- zasoby multimedialne zawarte w e-materiale;
- tablica interaktywna/tablica, pisak/kreda;
- oprogramowanie dla języka C++, w tym kompilator GCC/G++ 4.5 (lub nowszej wersji) i Code::Blocks 16.01 (lub nowszej wersji), Orwell Dev-C++ 5.11 (lub nowszej wersji) lub Microsoft Visual Studio.

Przebieg lekcji

Przed lekcją:

1. **Przygotowanie do zajęć.** Nauczyciel loguje się na platformie i udostępnia e-materiał: „Faktoryzacja w języku C++". Uczniowie zapoznają się z treściami w sekcji „Przeczytaj" w kontekście programowania.

Faza wstępna:

1. Nauczyciel wyświetla temat i cele zajęć. Prosi uczniów, by na podstawie wiadomości zdobytych przed lekcją zaproponowali kryteria sukcesu.
2. Prowadzący prosi uczniów, aby zgłaszali swoje propozycje pytań do tematu. Jedna osoba może zapisywać je na tablicy. Gdy uczniowie wyczerpią swoje pomysły, a pozostały jakieś ważne kwestie do poruszenia, nauczyciel je dopowiada.

Faza realizacyjna:

1. **Praca z tekstem.** Nauczyciel ocenia, na podstawie informacji na platformie, stan przygotowania uczniów do zajęć. Jeżeli jest ono niewystarczające, prosi wybraną osobę o przedstawienie najważniejszych informacji z sekcji „Przeczytaj".
2. **Praca z multimedium.** Nauczyciel wyświetla zawartość sekcji „Film samouczek". Uczniowie w parach wykonują polecenie 1, a następnie porównują swój program z przedstawionym w filmie.
   W kolejnym kroku uczniowie indywidualnie rozwiązują problem przedstawiony w poleceniu 3, a po ustalonym czasie prezentują na forum klasy wyniki swojej pracy.
3. **Ćwiczenie umiejętności.** Prowadzący zapowiada uczniom, że będą rozwiązywać ćwiczenie nr 1 z sekcji „Sprawdź się". Każdy z uczniów robi to samodzielnie. Po ustalonym czasie następuje porównanie napisanych kodów podczas wspólnego omówienia rozwiązań.
4. Liga zadaniowa – uczniowie pracując w parach, wykonują ćwiczenie nr 2 z sekcji „Sprawdź się", a następnie dzielą się swoimi wynikami przez porównywanie napisanego kodu z inną grupą, która również zakończyła zadanie.

Faza podsumowująca:

1. Nauczyciel wyświetla na tablicy temat lekcji i cele zawarte w sekcji „Wprowadzenie". W kontekście ich realizacji podsumowuje przebieg zajęć, a także wskazuje mocne i słabe strony pracy uczniów.
2. Na koniec zajęć z programowania w C++ nauczyciel prosi uczniów o rozwinięcie zdania: „Na dzisiejszych zajęciach nauczyłam/łem się jak…".

Praca domowa:

1. Uczniowie wykonują ćwiczenie 3 z sekcji „Sprawdź się".

Materiały pomocnicze:

- Oficjalna dokumentacja techniczna dla języka C++.
- Oficjalna dokumentacja techniczna dla kompilatora GCC/G++ 4.5 (lub nowszej wersji).
- Oficjalna dokumentacja techniczna dla oprogramowania Code::Blocks 16.01 (lub nowszej wersji), Orwell Dev-C++ 5.11 (lub nowszej wersji) lub Microsoft Visual Studio.

Wskazówki metodyczne:

- Film w sekcji „Film samouczek" można wykorzystać jako materiał przy powtórzeniu wiadomości.