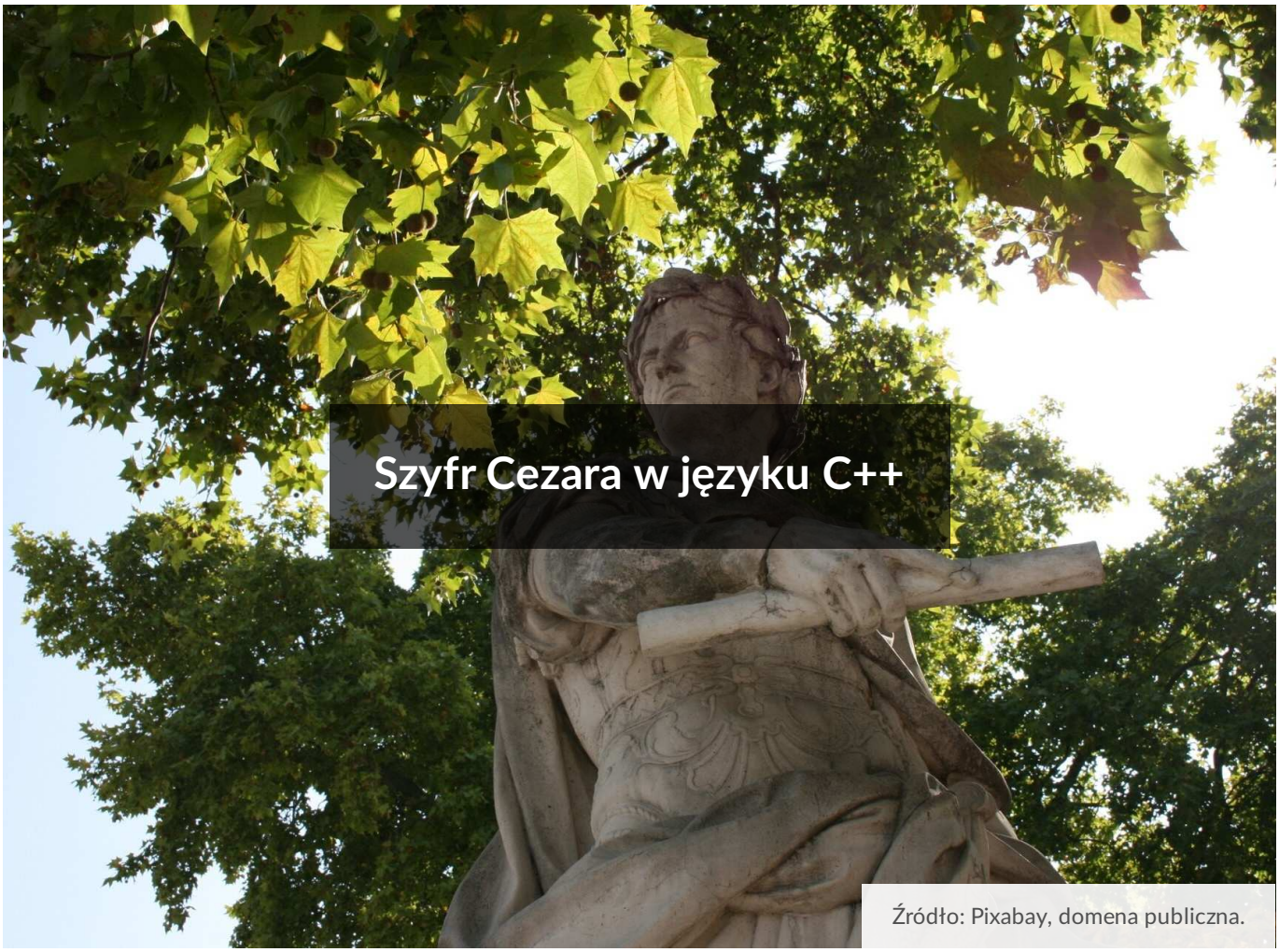
Szyfr Cezara w języku C++



Szyfr Cezara w języku C++

Źródło: Pixabay, domena publiczna.

Poznaliśmy już szyfr Cezara, czyli klasyczny szyfr przesuwający (podstawieniowy). Jego działanie polega na zastąpieniu każdej z liter tekstu jawnego odpowiadającą jej literą oddaloną o określoną liczbę miejsc w alfabecie.

W tym e-materiale dowiesz się, w jaki sposób można napisać algorytm szyfru Cezara w języku C++.

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

- Implementacja szyfru Cezara w języku Java,
- Szyfr Cezara w języku Python.

Więcej zadań? Sięgnij do Szyfr Cezara – zadania maturalne.

> **Twoje cele**
>
> - Utrwalisz wiedzę, na temat szyfru Cezara.
> - Przeanalizujesz komputerową realizację szyfru Cezara w języku C++.
> - Rozwiążesz kilka prostych zadań związanych z szyfrem Cezara.

Przeczytaj

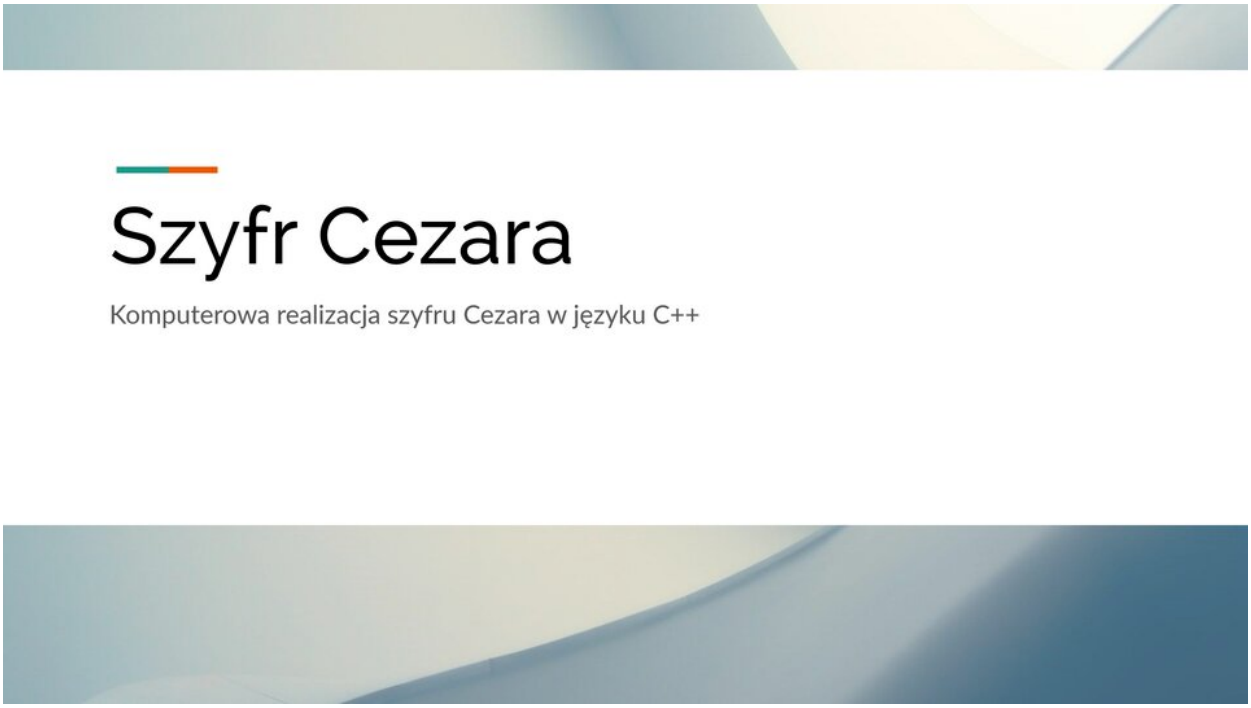
Problem 1

Specyfikacja problemu:

Napisz program, którego zadaniem będzie zaszyfrowanie i zdeszyfrowanie wiadomości „WIOSENNIE” dla klucza równego 3, przy użyciu szyfru Cezara.

Polecenie 1

Porównaj swoje rozwiązanie z animacją.



Film dostępny pod adresem /preview/resource/R1K4AZjNglIQr

Nagranie filmowe dotyczące szyfru Cezara - implementacji algorytmu szyfrowania i deszyfrowania wiadomości zapisanej alfabetem łacińskim.

Na wstępie warto przypomnieć sposób działania szyfru Cezara.

Każda litera z tekstu jawnego jest zamieniana na inną, oddaloną o określoną liczbę pozycji w alfabecie. Ta liczba liter zwana jest kluczem szyfrującym. Juliusz Cezar używał klucza równego 3. Zobaczmy zatem, w jaki sposób można zaszyfrować wszystkie litery alfabetu łacińskiego.

```
klucz = 3
alfabet łaciński     = ABCDEFGHIJKLMNOPQRSTUVWXYZ
alfabet zaszyfrowany = DEFGHIJKLMNOPQRSTUVWXYZABC
```

Z powyższego przykładu wynika, że litera `A` zamieniona została na literę `D`. Dzieje się tak, dlatego że litera `D` oddalona jest o 3 pozycje od litery `A` w alfabecie łacińskim.

Pseudokod

Pseudokod algorytmu szyfrującego przy użyciu szyfru Cezara wygląda następująco:

```
1 klucz = 3
2 tekst_jawny = "INFORMATYKA JEST SUPER"
3
4 dla i = 0, 1, ..., długość(tekst_jawny) wykonuj
5     zamień tekst_jawny[i] na oddaloną o klucz literę z alfabetu
```

W powyższym pseudokodzie wykorzystaliśmy następującą notację: `tekst_jawny[i]`. Ozncza ona odwołanie się od i-tego znaku w ciągu znaków. Pętla w linijce 4. wykonuje się, zaczynając od wartości 0, czyli założyliśmy, że indeks pierwszej litery z ciągu znaków wynosi 0 (jak w przypadku języka C++).

Komputerowa realizacja w C++

Zanim spróbujemy napisać własną implementację algorytmu szyfru Cezara, należy przypomnieć o kilku istotnych zagadnieniach związanych z językiem C++.

- Aby odwołać się do i-tego znaku w łańcuchu znaków (`string`), należy użyć notacji: `tekst_jawny[i]`, gdzie `tekst_jawny` to nazwa łańcucha znaków.
- W celu uzyskania długości łańcucha znaków użyjemy funkcji `length()` w następujący sposób: `tekst_jawny.length()`, gdzie `tekst_jawny` to nazwa łańcucha znaków.
- Wykonując dowolne działanie arytmetyczne na konkretnym znaku w łańcuchu znaków, należy pamiętać, że zostanie on zastąpiony kodem z tablicy ASCII. Przykładowo, w tablicy ASCII mała litera `a` ma wartość 97.

Jak to odszyfrować?

W celu zaszyfrowania liter należy wartość klucza szyfrującego dodawać do tekstu jawnego. Jeżeli chcemy odszyfrować tekst, wystarczy wartość klucza odejmować od tekstu zaszyfrowanego.

W prezentacji multimedialnej znajdziesz implementację algorytmu szyfrującego przy użyciu szyfru Cezara, napisaną w języku w C++.

Słownik

> **implementacja algorytmu**

> komputerowa realizacja algorytmu przy użyciu języka programowania; często mianem implementowania nazywamy również sam proces tworzenia programu
> **klucz szyfrujący**
>
> wartość liczbowa używana do określania liczby liter, o którą należy przesunąć litery z tekstu jawnego, aby uzyskać litery zaszyfrowane; np. jeśli klucz jest równy 3, to litera `A` zostanie zamieniona na literę `D`

Prezentacja multimedialna

Polecenie 1

Przeanalizuj prezentację, z której dowiesz się, w jaki sposób można dokonać komputerowej realizacji algorytmu szyfru Cezara w języku C++.

1

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/PA0z8YjzJ

Napiszemy program, który zaszyfruje poniższy tekst przy użyciu szyfru Cezara:

```
INFORMATYKA JEST SUPER
```

2

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/PA0z8YjzJ

Przyjmijmy, że będziemy posługiwać się wielkimi literami alfabetu łacińskiego. Składa się on z 26 liter. Szyfr Cezara ma klucz szyfrujący równy 3. Te dwie informacje są niezbędne do prawidłowej implementacji algorytmu.

3

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/PA0z8YjzJ

Możemy zatem utworzyć zmienne i wpisać do nich podstawowe wartości.

4

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/PA0z8YjzJ

Kolejnym krokiem będzie użycie pętli, aby przejść po każdej literze z tekstu do zaszyfrowania. Mamy możliwość sprawdzenia długości ciągu znaków, jaki będzie poddawany szyfrowaniu, więc możemy użyć pętli `for`. Pętla ta wykona się dokładnie `tekst.length()` razy. Skoro tekst ma długość 22, pętla wykona się 22 razy.

5

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/PA0z8YjzJ

Następnym elementem będzie wprowadzenie do pętli kodu, który umożliwi prawidłowe szyfrowanie tekstu. Teoretycznie moglibyśmy napisać to w ten sposób:

Jednak okazuje się, że powyższy kod nie jest uniwersalny. Dlaczego? Zastanów się chwilę, a następnie przejdź do kolejnego kroku.

6

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/PA0z8YjzJ

W kodzie z poprzedniego kroku są co najmniej dwa problemy. Pierwszy polega na tym, że

szyfrowaniu poddawane będą również znaki spacji. Oczywiście chcemy te znaki pominąć. Drugi problem to taki, że jeżeli będziemy chcieli zaszyfrować np. literę Z, której kod w tablicy ASCII wynosi 90, to po dodaniu wartości `klucz` otrzymamy kod 93. Znak z kodem ASCII równym 93 to:] (zamknięcie nawiasu kwadratowego). Jak wiemy, litera Z powinna być zaszyfrowana przy użyciu litery C. Poniżej znajdziesz propozycję, w jaki sposób można to osiągnąć.

Jak widzisz, wystarczyło dodać dwie instrukcje warunkowe. Pierwsza, która sprawdzi, czy w ogóle mamy do czynienia z literą, a druga, która sprawdzi, czy po dodaniu wartości zmiennej `klucz` wyjdziesz poza alfabet (otrzymana wartość będzie większa od kodu litery Z). W takiej sytuacji należy odjąć liczbę liter w alfabecie zapisaną w zmiennej `alfabet`.

7

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/PA0z8YjzJ

W ten oto sposób udało się nam napisać działającą implementację szyfru Cezara. Na zakończenie warto jeszcze dodać instrukcję wypisującą wartość `tekst`, aby zobaczyć zaszyfrowany tekst. Poniżej finalna wersja programu:

8

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/PA0z8YjzJ

Zastanów się, czy napisany program będzie działał prawidłowo, jeżeli zmienimy wartość zmiennej `klucz`?

9

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/PA0z8YjzJ

Odpowiedź brzmi: nie w każdym przypadku. Co się stanie, jeżeli podamy wartość `klucz = 50`? Nie ma tylu liter w alfabecie łacińskim. Jednak czy to problem? Gdy podana wartość klucza jest większa od 25, nastąpi zapętlenie i zaczniemy przechodzić po alfabecie po raz kolejny. Spójrz na przykład poniżej:

Skoro alfabet ma 26 liter, to przesunięcie litery A o 26 pozycji w alfabecie spowoduje przejście do końca alfabetu (aż do litery Z), a następnie powrót do litery A. Czyli klucz równy 26 jest tym samym co klucz równy 0, a klucz równy 27 to to samo co klucz równy 1.

10

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/PA0z8YjzJ

Spostrzeżenie z ostatniego kroku umożliwia nam udoskonalenie naszej implementacji przez dodanie instrukcji modulo (reszty z dzielenia) na klucz. Wystarczy dodać `% 26`, gdy inicjujemy wartość zmiennej `klucz`. Przykład: `int klucz = 30 % 26;`. W tym przypadku `klucz` równy 30 to to samo co `klucz` równy 4.

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/PA0z8YjzJ

Zatem ostateczna wersja naszego programu wygląda następująco:

Program wypisze zaszyfrowany tekst:
`LQIRUPDWBND MHVW VXSHU`

Sprawdź się

Pokaż ćwiczenia: 

Ćwiczenie 1

Napisz program, który zaszyfruje i wypisze tekst zapisany w zmiennej `tekst_jawny`, używając klucza ze zmiennej `klucz_szyfrujacy`.

Twoje zadania

1. Program szyfruje tekst zapisany w łańcuchu znaków `tekst_jawny`.

Ćwiczenie 2

Napisz program, który zaszyfruje i wypisze tekst zapisany w zmiennej `tekst_jawny`, używając klucza ze zmiennej `klucz_szyfrujacy`. Uwaga! Tekst jawny zawiera spacje. Pamiętaj, aby ich nie szyfrować.

Twoje zadania

1. Program szyfruje tekst zapisany w łańcuchu znaków `tekst_jawny`.

Ćwiczenie 3

Napisz program, który odszyfruje łańcuch znaków `tekst_zaszyfrowany` przy użyciu klucza szyfrującego `klucz`.

Twoje zadania

1. Program odszyfruje tekst zapisany w łańcuchu znaków `tekst_zaszyfrowany`.

Dla nauczyciela

Autor: Maurycy Gast
Przedmiot: Informatyka

Temat: Szyfr Cezara w języku C++

Grupa docelowa:

Szkoła ponadpodstawowa, liceum ogólnokształcące, technikum, zakres podstawowy i 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.

Treści nauczania – wymagania szczegółowe

I. Rozumienie, analizowanie i rozwiązywanie problemów.

Zakres podstawowy. Uczeń:

2) stosuje przy rozwiązywaniu problemów z różnych dziedzin algorytmy poznane w szkole podstawowej oraz algorytmy:

b) na tekstach: porównywania tekstów, wyszukiwania wzorca w tekście metodą naiwną, szyfrowania tekstu metodą Cezara i przestawieniową,

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):

- Utrwalisz wiedzę, na temat szyfru Cezara.
- Przeanalizujesz komputerową realizację szyfru Cezara w języku C++.
- Rozwiążesz kilka prostych zadań związanych z szyfrem Cezara.

Strategie nauczania:

- konstruktywizm;
- konektywizm.

Metody i techniki nauczania:

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

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: „Szyfr Cezara w języku C++". Uczniowie zapoznają się z multimedium w sekcji „Prezentacja multimedialna".

Faza wstępna:

1. **Rozpoznanie wiedzy uczniów.** Nauczyciel zadaje uczniom pytania dotyczące ich aktualnego stanu wiedzy w obszarze poruszanego tematu i programowania, np.
   – jaka jest zasada działania szyfru Cezara?
   – czym jest klucz szyfrujący?
   Chętni uczniowie udzielają na nie odpowiedzi.

Faza realizacyjna:

1. Uczniowie analizują przykład z sekcji „Przeczytaj" i powtarzają zaprezentowane rozwiązanie na swoim komputerze.

2. **Praca z multimedium.** Nauczyciel wyświetla zawartość sekcji „Prezentacja multimedialna”, wybrany uczeń czyta treść polecenia nr 1: „Przeanalizuj prezentację, z której dowiesz się, w jaki sposób można dokonać komputerowej realizacji algorytmu szyfru Cezara w języku C++.” i omawia przykładowe rozwiązanie postawionego problemu.
3. **Ćwiczenie umiejętności.** Nauczyciel przechodzi do sekcji „Sprawdź się”. Uczniowie indywidualnie rozwiązują ćwiczenia nr 1-2 na czas. Osoba, która poprawnie rozwiąże zadania jako pierwsza, wygrywa, a nauczyciel może nagrodzić ją oceną za aktywność.

Faza podsumowująca:

1. Nauczyciel ponownie wyświetla na tablicy temat lekcji zawarty w sekcji „Wprowadzenie” i inicjuje krótką rozmowę na temat zrealizowanych celów (czego uczniowie się nauczyli).
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:

- Uczniowie mogą wykorzystać multimedium w sekcji „Prezentacja multimedialna” do przygotowania się do lekcji powtórkowej.