Politechnika Śląska
Wydział Matematyk Stosowanej
Kierunek Informatyka

Gliwice, 25.01.2023

Programowanie I

projekt zaliczeniowy

"*Szyfrator*"

Karolina Kozubik gr. lab. 3/5

1. Opis projektu.

Aplikacja przeznaczona do szyfrowania krótkich wiadomości tekstowych za pomocą prostych szyfrów: szyfru Cezara, szyfru Vigenère'a, szyfru „cztery kwadraty" oraz szyfru płotkowego.

2. Wymagania

- szyfrowanie i deszyfrowanie szyfrem Cezara,
- szyfrowanie i deszyfrowanie szyfrem Vigenère'a,
- szyfrowanie i deszyfrowanie szyfrem „cztery kwadraty",
- szyfrowanie i deszyfrowanie szyfrem płotkowym,
- opcja zapisywania wyniku operacji do pliku,
- interfejs użytkownika w języku polskim i angielskim

3. Przebieg realizacji

Pliki programu:

Program można podzielić na dwie główne części: część odpowiedzialną za szyfrowanie oraz część odpowiedzialną za interfejs użytkownika.

Pierwsza część programu zawiera klasy odpowiedzialne za szyfrowanie (wraz ze strukturami i enumeratorami, z których korzystają te klasy), umieszczone w plikach: *CipherStrategy.h, Ceasar.h, CeasarCipher.h, CeasarCipher.cpp, four_square.h, FourSquareCipher.h, FourSquareCipher.cpp, RailFenceCipher.h, RailFenceCipher.cpp, VigenereCipher.h, VigenereCipher.cpp*. Plik *CipherStrategy.h* zawiera klasę abstrakcyjną CipherStrategy, a w pozostałych wymienionych powyżej plikach znajdują się klasy po niej dziedziczące, implementujące odpowiednie szyfry. Poza tym, część programu odpowiedzialna za szyfrowanie zawiera klasę CipherCommunicator odpowiedzialną za komunikację z interfejsem użytkownika, umieszczoną w plikach *CipherCommunicator.h, CipherCommunicator.cpp*. Plik *ciphers.h* zawiera enumeratory dostępnych szyfrów oraz możliwych akcji (szyfrowanie, deszyfrowanie). W plikach *EncryptionContext.h, EncryptionContext.cpp* umieszczona jest klasa EncryptionContext, odpowiedzialna za przechowywanie i używanie aktualnie wybranej klasy szyfru.

Część programu odpowiedzialna za interfejs użytkownika zawiera klasę abstrakcyjną UserInterface wraz z implementacją interfejsu użytkownika (klasa ConsoleUI) w plikach *UserInterface.h, ConsoleUI.h, ConsoleUI.cpp*. Plik *Communicator.h* zawiera klasę Communicator. Klasy dziedziczące po klasie Communicator mają możliwość korzystania z klas interfejsu użytkownika. Plik *screens.h* zawiera klasę Screen oraz enumerator ScreenType. Klasy dziedziczące po klasie Screen umieszczone są w plikach *MainScreen.h, MainScreen.cpp, SettingsScreen.h, SettingsScreen.cpp*. Klasy te są odpowiedzialne za wywoływanie odpowiednich funkcji w odpowiedzi na interakcje użytkownika. Klasa ScreenController umieszczona w plikach *ScreenController.h, ScreenController.cpp* jest odpowiedzialna za zarządzanie kolejnością ekranów wyświetlanych użytkownikowi. Pliki *ScreenCommunicator.h, ScreenCommunicator.cpp* zawierają klasę ScreenCommunicator. Klasy dziedziczące po tej klasie odpowiadają za komunikację między odpowiednimi ekranami oraz interfejsem użytkownika i umieszczone są w plikach *MainScreenCommunicator.h, MainScreenCommunicator.cpp, SettingsScreenCommunicator.h, SettingsScreenCommunicator.cpp*. Plik *lang.h* zawiera enumeratory Language oraz LangCode. Pliki *en.h, pl.h* zawierają tłumaczenia komunikatów wyświetlanych na ekran.

Pliki *validators.h, validators.cpp* zawierają funkcje odpowiedzialne za walidację danych przekazywanych przez użytkownika do programu. Pliki *FileManager.h, FileManager.cpp* zawierają klasę FileManager odpowiedzialną za pracę z plikami (w szczególności za zapisywanie wyniku szyfrowania / deszyfrowania do pliku). Plik *Settings.h* zawiera strukturę Settings przechowującą ustawienia użytkownika. Pliki *utils.h, utils.cpp* zawierają funkcje przeprowadzające podstawowe operacje na używanych w programie strukturach danych. Pliki *Exception.h, FileException.h, ValidationException.h* zawierają klasy wyjątków z których korzysta program. Plik *main.cpp* zawiera funkcję main.

Szyfr Cezara:

Szyfrowanie może być reprezentowane przez wyrównanie dwóch alfabetów; alfabet szyfrujący jest alfabetem zwykłym obróconym w lewo lub w prawo o pewną liczbę pozycji. Na przykład, oto szyfr Cezara wykorzystujący obrót w lewo o trzy miejsca, co odpowiada przesunięciu w prawo o 23:

Zwykły alfabet: ABCDEFGHIJKLMNOPQRSTUVWXYZ

Alfabet szyfrujący: XYZABCDEFGHIJKLMNOPQRSTUVW

Algorytm szyfrowania polega na zamianie kolejnych liter zwykłego alfabetu na litery alfabetu szyfrującego na tej samej pozycji. Deszyfrowanie polega na zamianie w drugą stronę

Szyfr Vigenère'a:

Szyfr Vigenère'a wykorzystuje słowo kluczowe. Do słowa kluczowego dodawane są jego kolejne litery, aż jego długość będzie równa długości wiadomości. Na przykład dla wiadomości „programowanie" słowo kluczowe „example" zostanie następująco przekształcone:

programowanie

exampleexampl

Następnie każda z liter wiadomości szyfrowana jest szyfrem Cezara. Używane jest przesunięcie, dla którego pierwszą literą alfabetu szyfrującego jest litera słowa kluczowego odpowiadająca obecnie szyfrowanej literze. Deszyfrowanie polega na deszyfrowaniu szyfrem Cezara (zamiana w drugą stronę).

Szyfr Cztery Kwadraty:

Szyfr „Cztery Kwadraty" wykorzystuje cztery matryce 5 na 5 ułożone w kwadrat. Każda z matryc 5 na 5 zawiera litery alfabetu (pomijając "Q). Górna lewa i dolna prawa matryca zawierają standardowy alfabet. Prawa górna i lewa dolna matryca rozpoczyna się od podanego słowa kluczowego, z pominięciem powtarzających się znaków. Następnie dodawany jest alfabet bez liter, które znajdują się w słowie kluczowym.

Algorytm:

- podzielenie tekstu do zaszyfrowania w pary
- znalezienie pierwszej litery w parze w lewej górnej matrycy
- znalezienie drugiej litery w parze w prawej dolnej matrycy
- pierwsza zaszyfrowana litera znajduje się w prawej górnej matrycy, w tym samym rzędzie co pierwsza litera w parze, i w tej samej kolumnie co druga litera w parze
- druga zaszyfrowana litera znajduje się w lewej dolnej matrycy, w tym samym rzędzie co druga litera w parze, i w tej samej kolumnie co pierwsza litera w parze

Na przykład, szyfrując tekst ze słowami kluczowymi *example* i *keyword*, cztery matryce wyglądają następująco:

```
a b c d e   e x a m p
f g h i j   l b c d f
k l m n o   g h i j k
p r s t u   n o r s t
v w x y z   u v w y z


k e y w o   a b c d e
r d a b c   f g h i j
f g h i j   k l m n o
l m n p s   p r s t u
t u v x z   v w x y z
```

Zaszyfrowany tekst „programowanie” będzie wyglądał następująco: „olhcnekhuejbe”

Rozszyfrowanie działa w ten sam sposób, ale w odwrotnej kolejności. Litery z pary są szukane w prawej górnej i lewej dolnej matrycy, a odszyfrowane litery znajdują się w ich rzędach i kolumnach, kolejno w lewej górnej i prawej dolnej matrycy.

Szyfru Płotkowy:

Szyfr płotkowy korzysta z liczby rzędów (n) na wymyślonym płotku, i rozmieszcza kolejne litery szyfrowanej wiadomości na rzędach tam i z powrotem. Na przykład dla n = 3, szyfrowanie wiadomości „programowanie” wygląda następująco:

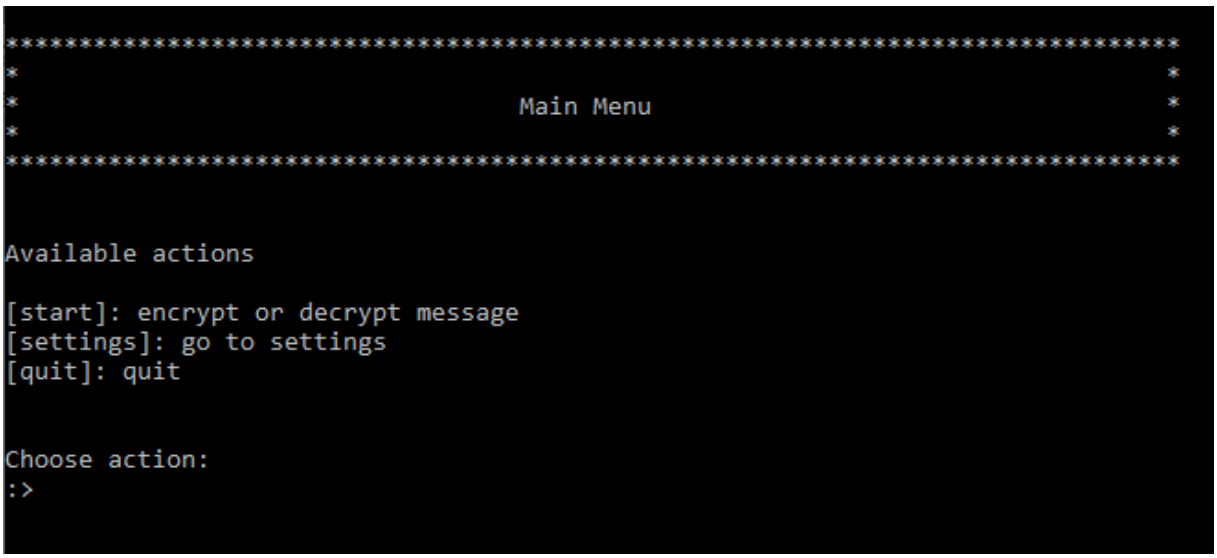
```
p   r   w   e
 r g a o a i
  o   m   n
```

Następnie przepisywane są słowa z każdego rzędu, także ostateczny rezultat szyfrowania wiadomości „programowanie” wygląda następująco: „prwe rgaoai omn”. Deszyfrowanie polega ponownym rozmieszczeniu zaszyfrowanych słów na „rzędach płotka” i odczytaniu ich w odpowiedniej kolejności.

4. Instrukcja użytkownika

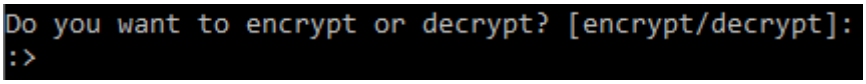
Interakcje z użytkownikiem:

Za każdym razem, gdy program prosi użytkownika o wpisanie danych, zostanie pokazany symbol: :> W komunikacie zawarte będą opcje możliwe do wybrania, umieszczone w nawiasach kwadratowych. Na przykład przy ekranie głównym, poniższy komunikat że należy wpisać jedno ze słów: „start”, „settings”, „quit”.



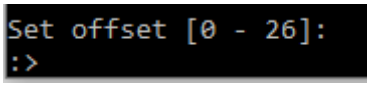
Rysunek 1: Ekran główny programu

Jeśli w nawiasach kwadratowych umieszczone jest kilka opcji oddzielonych ukośnikiem, należy wybrać jedną z nich. Na przykład przy wyborze akcji do przeprowadzenia pokazuje się komunikat:

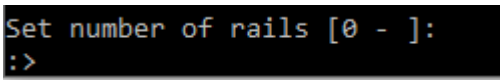


Rysunek 2: wybór operacji do przeprowadzenia

Jeśli w nawiasach kwadratowych umieszczone dwie liczby oddzielone myślnikiem, należy wybrać liczbę z danego zakresu. Jeśli jednej z liczb brakuje, nie ma ograniczenia z danej strony zakresu.



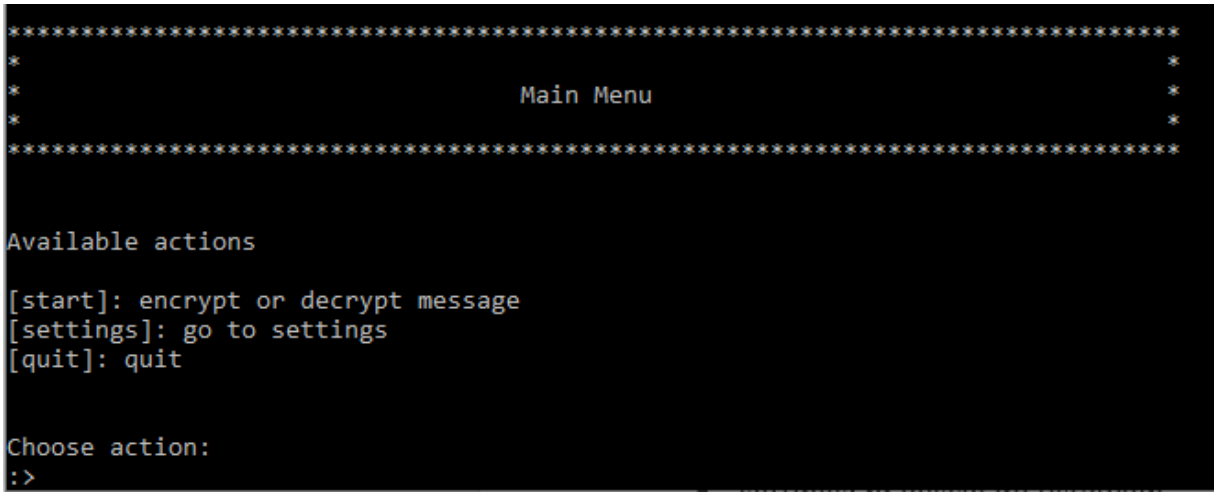
Rysunek 3: Oczekiwana jest liczba z zakresu od 0 do 26



Rysunek 4: Oczekiwana liczba większa od 0

Przebieg działania programu:

Po rozpoczęciu działania programu prezentowany jest ekran główny:



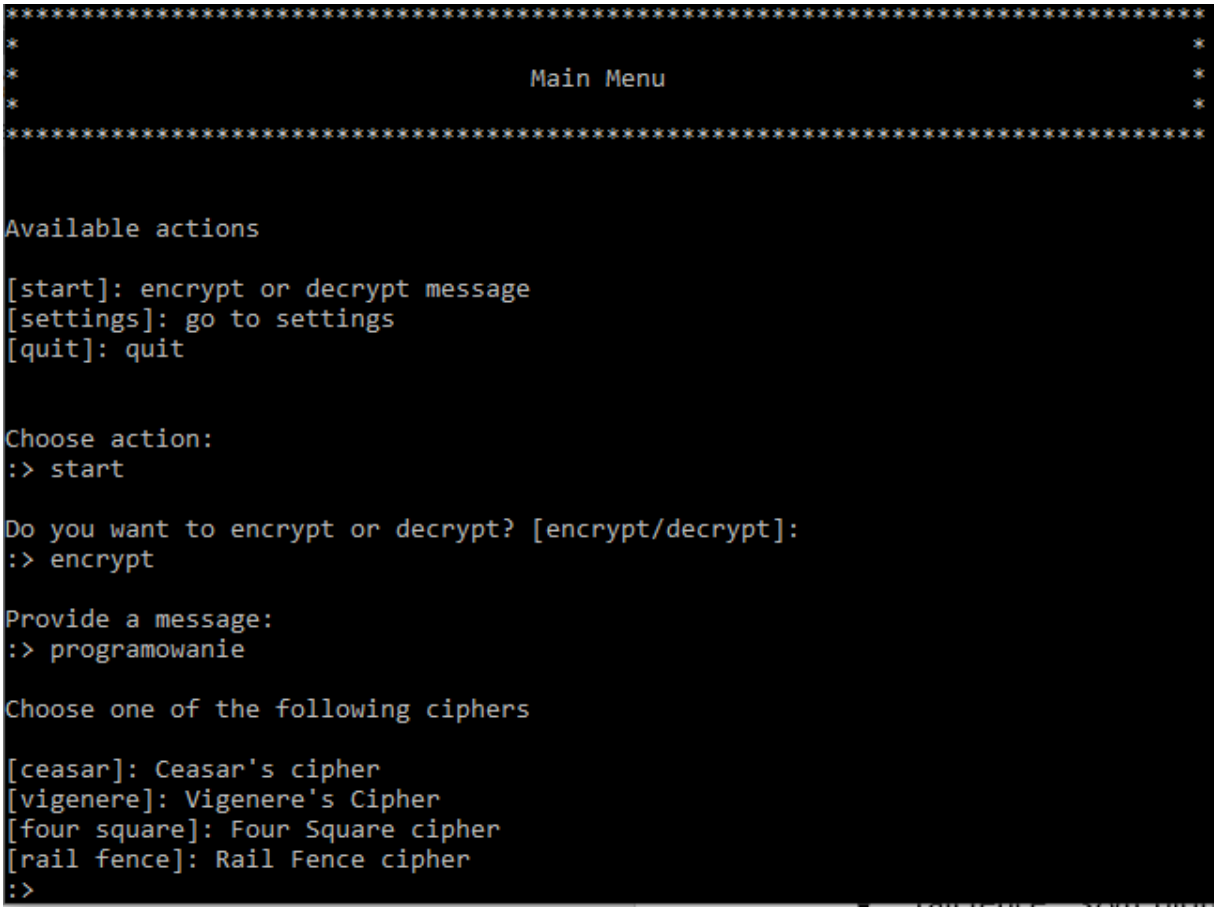
Rysunek 5: Ekran główny programu

Użytkownik może wybrać jedną z trzech opcji:

- start: rozpoczęcie szyfrowania lub deszyfrowania,
- settings: przejście do ustawień,
- quit: zakończenie działania programu

Po wybraniu opcji „start” program prosi użytkownika o wybranie akcji do zrealizowania: szyfrowanie lub deszyfrowanie (odpowiednio: „encrypt”, „decrypt”). Po wybraniu akcji program prosi użytkownika o podanie wiadomości do zaszyfrowania/odszyfrowania. Następnie program prosi o wybranie jednego z dostępnych szyfrów:

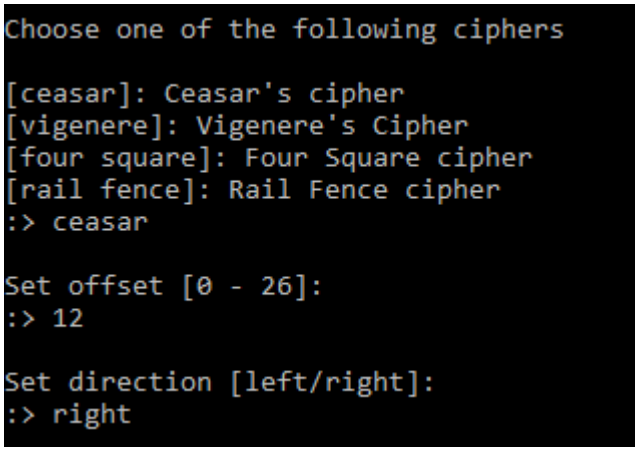
- „ceasar”: szyfr Cezara,
- „four square”: szyfr „cztery kwadraty”,
- „vigenere”: szyfr Vigenère’a,
- „rail fence”: szyfr płotkowy



Rysunek 6: Widok po wybraniu opcji "start", "encrypt" i podaniu wiadomości "programowanie"

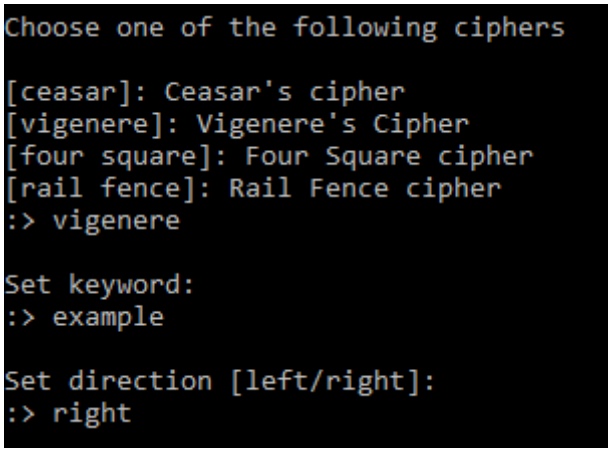
Następnie program poprosi o podanie dodatkowych danych w zależności od wybranego szyfru.

Szyfr Cezara: Program prosi o podanie przesunięcia (offset). Przesunięcie powinno być liczbą większą od 0 i mniejszą od 26. Następnie program poprosi o podanie kierunku (direction). Należy podać jedną z możliwych opcji: „left” lub „right”. Program rozpocznie szyfrowanie wiadomości.



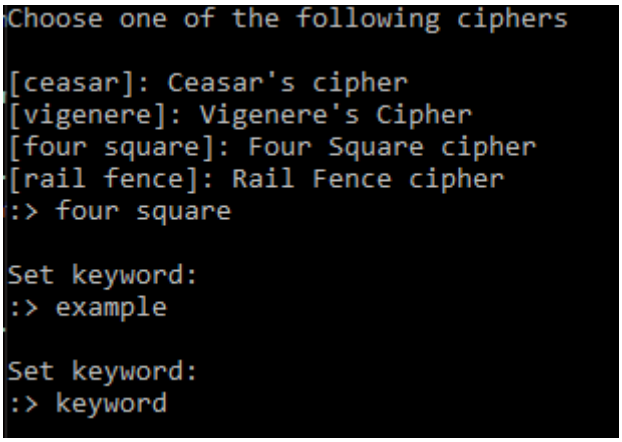
Rysunek 7: Ustawienia szyfru Cezara

Szyfr Vigenère’a: Program prosi o podanie słowa kluczowego. Następnie program poprosi o podanie kierunku (direction). Należy podać jedną z możliwych opcji: „left” lub „right”. Program rozpocznie szyfrowanie wiadomości.



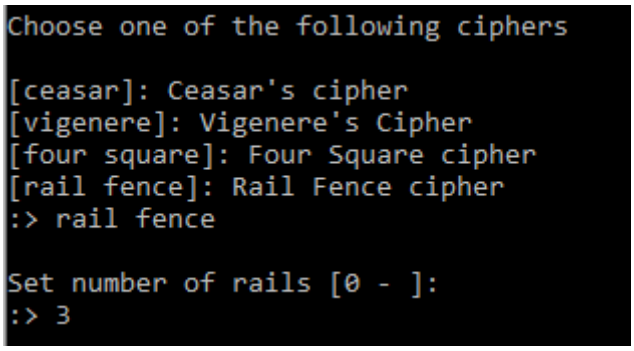
Rysunek 8: Ustawienia szyfru Vigenère’a

Szyfr „cztery kwadraty”: Program dwukrotnie poprosi o podanie słowa kluczowego. Słowa kluczowe mogą być takie same, ale działanie szyfru jest mocniejsze, jeśli podane zostaną różne słowa. Program rozpocznie szyfrowanie wiadomości.



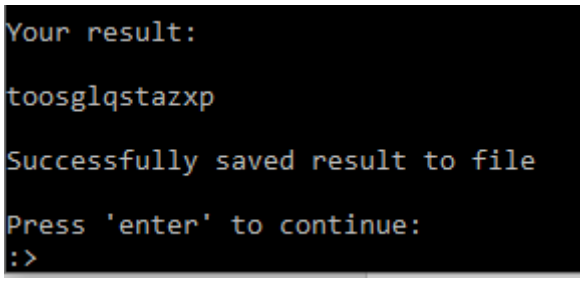
Rysunek 9: Ustawienia szyfru "cztery kwadraty"

Szyfr płotkowy: Program prosi o podanie liczby „sztachet”, która powinna być większa od 0. Program rozpocznie szyfrowanie wiadomości.



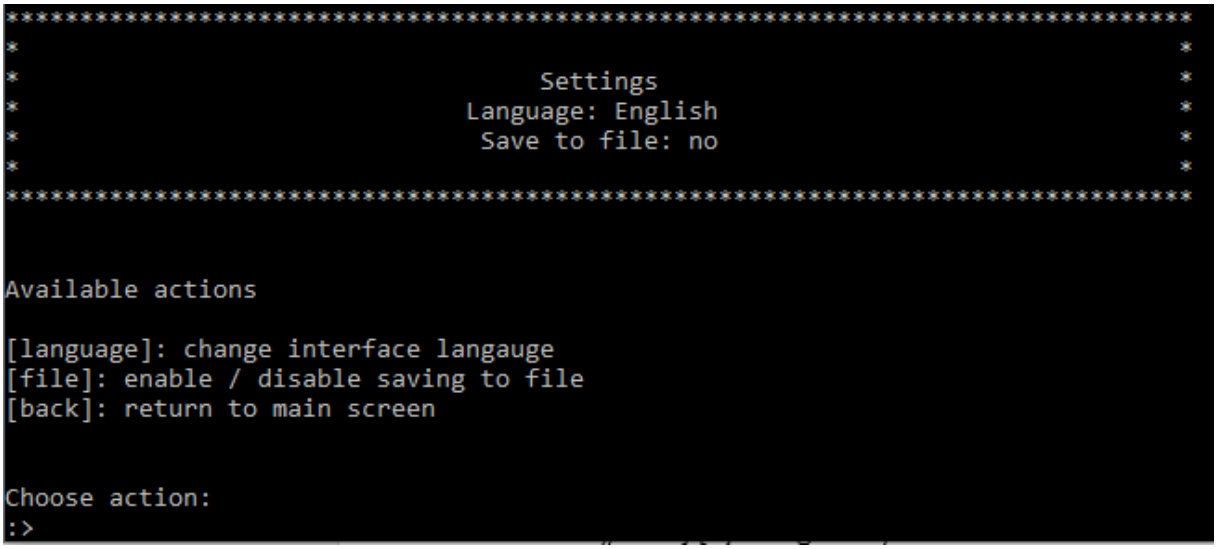
Rysunek 10: Ustawienia szyfru płotkowego

Po zaszyfrowaniu wiadomości program wypisze wynik na ekran. Jeśli ustawiona została opcja zapisu do pliku, program wygeneruje plik tekstowy z wynikiem. Następnie program zaczeka, aż użytkownik przyciśnie klawisz „enter” i przejdzie do ekranu głównego.



Rysunek 11: Komunikat wyświetlany po zaszyfrowaniu wiadomości

Po wybraniu opcji „settings” w ekranie głównym program przejdzie do ekranu ustawień.



Rysunek 12: Ekran ustawień

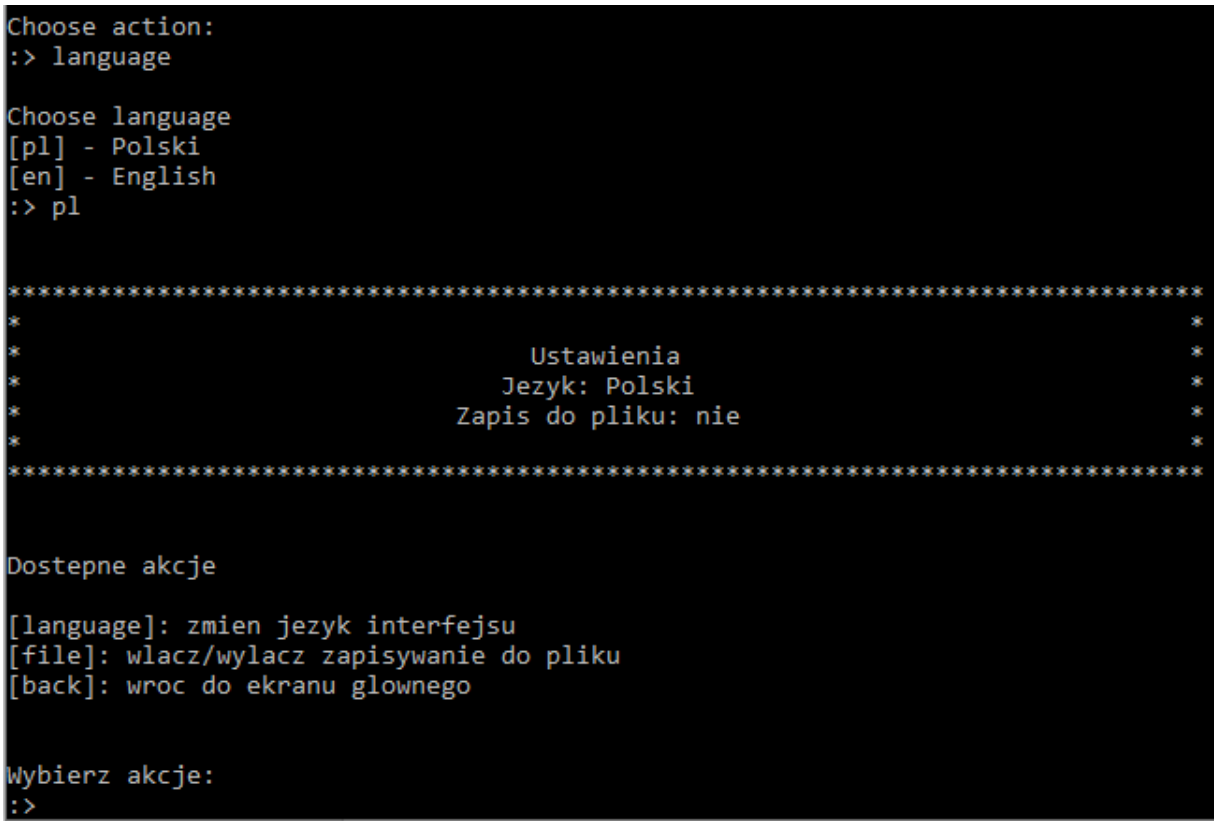
Pod tytułem ekranu widoczne są obecnie ustawione ustawienia.

Użytkownik może wybrać jedną z trzech opcji:

- language: zmień język interfejsu użytkownika,
- file: włącz/wyłącz opcję zapisu do pliku
- back: wróć do ekranu głównego

Po wybraniu opcji „language” program prosi o wybranie jednego z dostępnych języków:

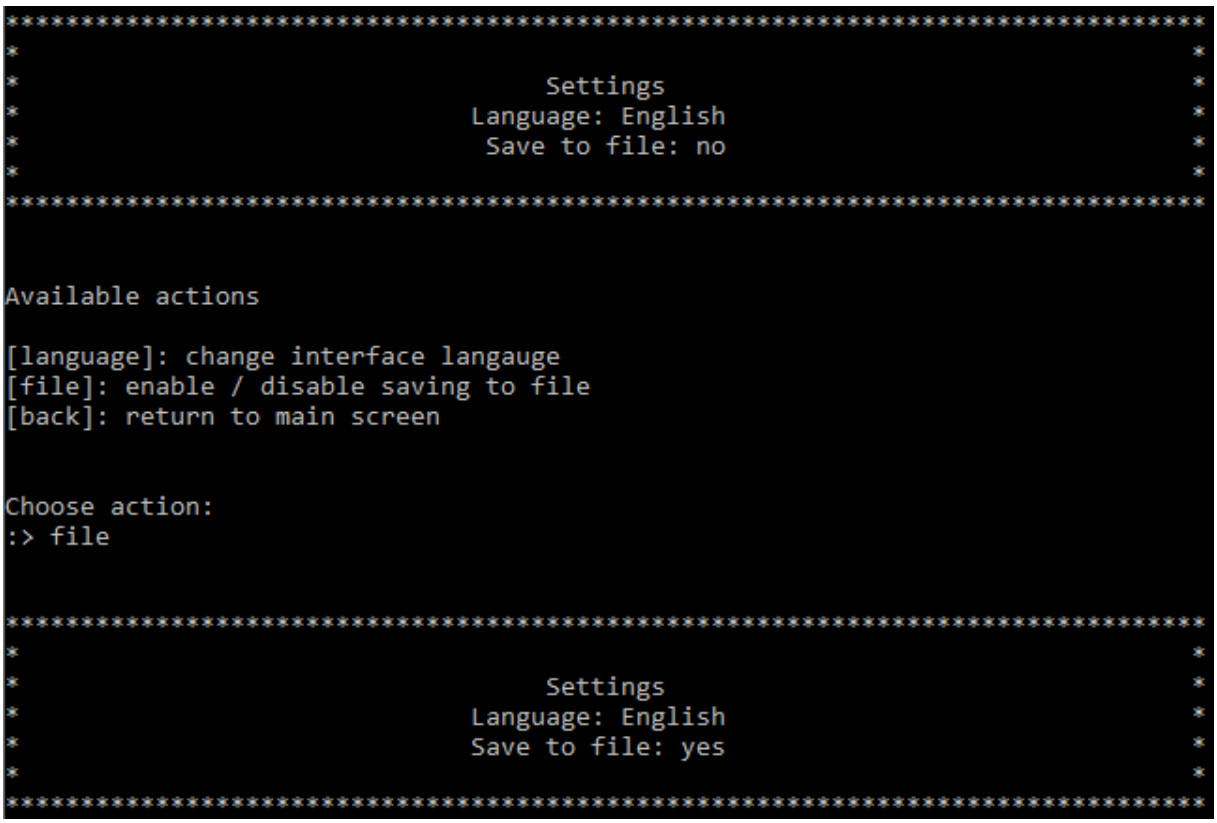
- „en”: język angielski,
- „pl”: język polski



Rysunek 13: Zmiana języka interfejsu użytkownika

Następnie program zmieni język interfejsu użytkownika.

Po wybraniu opcji „file” program zmieni ustawienie opcji zapisu do pliku.



Rysunek 14: Zmiana opcji zapisu do pliku

Po wybraniu opcji „back” program powróci do ekranu głównego.

Po wybraniu opcji „quit” w ekranie głównym program zakończy swoje działanie.

Jeśli użytkownik poda nieprawidłowe dane wyświetlony zostanie odpowiedni komunikat, a program poprosi o ponowne podanie danych.

5. Podsumowanie i wnioski.

Program spełnia wszystkie założenia projektu. Ponadto program waliduje dane podawane przez użytkownika. Program używa wzorca projektowego „strategia”. W przyszłości można dodać funkcjonalność zapisywania obecnych ustawień do pliku, tak aby zostały one zachowane przy ponownym uruchomieniu programu, oraz opcję wyświetlenia informacji na temat każdego z wymienionych szyfrów.