1. (za 2 punkty) Maszyna szyfrująca Enigma używana przez armię niemiecką w czasie drugiej wojny światowej składała się z wstępnej łącznicy, trzech wirników i pierścienia odwracającego. Każdy z tych elementów realizował permutację 26 dużych liter alfabetu łacińskiego (maszyna nie obsługiwała ani małych liter ani liter specyficznych dla języka niemieckiego). Litery były wprowadzane do maszyny przez naciśnięcie klawisza, co podawało prąd na jedną z 26 linii, następnie prąd wędrował przez łącznicę wstępną i wirniki docierając do pierścienia odwracającego. Pierścień odwracający łączył ze sobą pary przewodów, tak że po przejściu przez pierścień odwracający prąd wracał inną linią ponownie przez wirniki i łącznicę wstępną docierając w końcu do żaróweczek. Zapalona żaróweczka pozwalała odczytać wynik szyfrowania. Po zaszyfrowaniu litery następowało przesunięcie wirników (jednego lub czasami więcej). Logicznie łącznica wstępna realizowała permutację zadaną przez operatora maszyny (z pewnymi ograniczeniami które tu pominiemy), każdy wirnik realizował ustaloną (przez połączenia elektryczne) permutację, pierścień odwracający realizował permutację która była iloczynem cykli długości 2. Przepływ prądu z powrotem oznaczał wykonanie permutacji odwrotnej do permutacji robionej przez łącznicę wstępną i wirniki. Ruch wirników powodował że efektywnie między wirnikami były wprowadzone dodatkowe permutacje rotujące alfabet.

Celem zadania jest zasymulowanie szyfrowania w podobnym stylu, jednakże z pewnymi uproszczeniami. Mianowicie system szyfrujący ma się składać z czterech klas. Jedna klasa ma być abstrakcyjna, tzn. ma specyfikować metody. Druga klasa to ma być szyfr trywialny czyli identyczność. Trzecia i czwarta klasa mają mieć dodatkowe pole wskaźnika na klasę szyfrującą. Dokładniej, metoda szyfrująca znak tych klas po wykonaniu własnej pracy ma przekazać pośredni wynik do metody szyfrującej klasy dostępnej przez wskaźnik i zwracać wynik z tej klasy. Taka konstrukcja oznacza że możemy połączyć klasy szyfrujące w łańcuch. Oczywiście, by uniknąc nieskończonej rekursji na końcu łańcucha musimy mieć klasę która nie woła kolejnego kroku, do tego przydaje się klasa trywialna. Trzecia klasa ma realizować zadaną w kontruktorze permutację (i wołać następny etap). Czwarta klasa ma realizować rotację alfabetu (i też wołać następny etap). Dokładniej, rotacja ma dodawać wielkość `l` do kodu znaku (kody większe niż maksywmalny mają się zawinąć na początek, czyli jest to dodawanie modulo). Po każdym wywołaniu ta klasa powiększa `l` o stałą wartość `k`. Różne szczegóły pozostawiam do wyboru, szyfrowanie może działać na dużych literach lub na bajtach. Kluczowe jest by kombinując wywołania konstruktorów można było zestawić łańcuch z permutacji (tych ustalonych i rotacji) i by zestawiony łańcuch szyfrował zgodnie z opisem wyżej.

Komentarz: By wiernie symulować Enigmę trzeba by zestawić odpowiedni łańcuch i dobrać (zmienić) regułę zmian `l`-ów by się zgadzała z tym co robiła Enigma.

2. (łącznie za 2 punkty) Napisz analizator syntaktyczny bazujący na priorytetch dla wyrażeń reprezentujący symbole jako odpowiednią klasę, produkujący drzewo wyrażenia. Same symbole mogą pochodzić ze skanera z zadania 2 z listy 12. Analizator syntaktyczny można otrzymać przez modyfikację programu przykładowego `rpn.cpp`. By otrzymać drzewo trzeba mieć wspólną klasę która pozwala na przechowywanie zarówno symboli jak i drzew. Dla argumentów operatorów trzeba tworzyć proste drzewa (liście) i umieszczać je na stosie (to jest różnca w porównaniu z programem przykładowym). Druga różnica to postać stosu, będą na nim na przemian operatory i (pod)drzewa. A więc do porównania priorytetu operatora na stosie z priorytetem bierzącego operatora trzeba by zaglądać poniżej pierwszego elementu stosu. Prościej jest pamiętać prorytet operatora na stosie w dodatkowej zmiennej. Analizator syntaktyczny jest za 1.4 punktu.

W drugiej części połącz analizator syntaktyczny z procedurą obliczania wartości wyrażenia reprezentowanego przez drzewo (z zadania 1 z listy 11) tak by łącznie otrzymać kalkulator obliczający wartości wyrażeń (to jest za 0.6 punktu)