Софийски университет „Св. Климент Охридски“
Факултет по математика и информатика

Контролно 1

курс Структури от данни и програмиране
за специалност Компютърни науки, поток 2
зимен семестър 2021/2022 г.

1. Основните критерии при оценяването на контролните ще бъдат:
    - успешно изпълнение на поставеното условие;
    - използването на най-подходящите структури от данни;
2. Другите критерии при оценяването са:
    - добро стилизиране и форматиране на кода;
    - сложности;
    - следване на добри практики за писане на код;
    - спазване на ООП парадигмата

Задача 1. Нека е даден следният шаблон на структура:

```
template <typename T>
struct Node {T data; Node<T> *next;};
```

При това условие, нека е даден списък L с елементи стекове. Възлите на L са от тип `Node<std::stack<T>>`. Всеки стек може да съдържа различни елементи (**числа**). Някои стекове могат да съдържат само четни, други само нечетни, а трети четни и нечетни елементи. Казваме, че два стека са подобни, ако съдържат само четни или само нечетни елементи. Да се дефинира подходящо параметризирана функция `concatSimilarStackElements(L)`, която конкатенира всички последователни стекове, които са подобни и връща указател към новия свързан списък.

Пример: Даден е списък от стекове. Първите два стека са с четни числа. Затова вторият се конкатенира в началото на първия, запазвайки реда на добавяне на елементите. Третият и четвъртия стек са с нечетни числа и те също се конкатенират. Петият е с четни числа, но преди него и след него няма други стекове с четни елементи и той остава непроменен. Последния стек съдържа различни елементи и също не се конкатенира.

върхове	върхове
6 **4** **0** 7 1 **0** 2 **2** → **8** → 5 → 9 → **2** → 1	**0** **8** 1 **6** 9 **4** 7 **0** 2 **2** → 5 → **2** → 1

Задача 2.
Разглеждаме езика OYAML (“Oversimplified YAML”), опростен вариант на езика YAML (“Yet Another Markup Language”), което е човешки-четим език за сериализация на данни.

OYAML файловете се речници, съставени от двойки ключ-стойност. Ключовете са низове, състоящи се само от малки латински букви, а съответните им стойности са два вида - прости и съставни. Простите стойности са низове с произволна дължина, но без нови редове, а сложните - цели съставни речници.

В OYMAL формата ключовете винаги са следвани от двоеточие и интервал, след което е стойността на ключа, ако е проста. Ако стойността е съставна, вложеният речник започва от следващия ред.

Ключове са предхождани от интервали. Броят на интервалите е равен на нивото на влагане на речника. В следния пример:

```
dictName: main dictionary
value: 5
innerDict1:
  dictName: first inner dictionary
  value: 10
  anotherValue: 100
innerDict2:
  dictName: second inner dictionary
  value: 15
  purpose: show more levels of nesting
  innerDict3:
    dictName: innermost dictionary
    value: 20
    someOtherKey: anything here
```

Речникът с име “main dictionary” е основен за файла и е с ниво на влагане 0. “first inner dictionary” и “second inner dictionary” са с вложени в “main dictionary”, а “innermost dictionary” е вложен в “second inner dictionary”.

Дефинирайте тип данни YAMLDocument, подходящ за представяне в паметта на данните от OYAML файл. Да се поддържат следните операции:

- `insert(locatorKey, newKey, newValue)`: измежду всички вложени речници на всички нива, намира някой речник, съответен на ключ locatorKey и добавя към него нова двойка ключ-проста стойност. Например,
  `insert(“innerDict3”, “test”, “5”)`
  ще вмъкне “test: 5” в “innermost dictionary”.
  Ако такъв речник не е намерен (и в частност, ако ключът за търсене е празен низ), се вмъква в главния речник
- `find(path)`, където path е символен низ от вида “`<key1>/<key2>/.../<keyN>`”. Ако key1, key2,...,keyN-1 са последователно вложени един в друг речници, а keyN има проста стойност, да се върне стойността на keyN. В противен случай да се върне празен низ. Например:
  `find(“innerDict2/innerDict3/value”)`
  ще върне низа “20”, а
  `find(“value”)`
  ще върне низа “5”