

UNIVERZITET U TUZLI
FAKULTET ELEKTROTEHNIKE



Strukture podataka

Zadaća 1

Tuzla, mart/ožujak 2020.

Sadržaj

Sadržaj	2
Zadatak 1	3
Zadatak 2	3
Način predavanja	5

Zadatak 1

Na kod koji se nalazi u sklopu laboratorijski vježbi 4 (lv4.zip u mapi predmeta) izgraditi sljedeće funkcionalnosti:

- implementirati klase `iterator` i `const_iterator`, kao i sve potrebne metode kako bi lista implementirana nizom (`ArrayList`) podržavala rad sa iteratorima tipa `random_access`,
- implementirati metode `begin()` i `end()` kao i njihove verzije koje vraćaju `const_iterator`: `cbegin()` i `cend()`.
- implementirati metod `insert(it, value)` koji na poziciju gdje pokazuje iterator `it` dodaje element `value` pri čemu metod ne vraća nikakvu povratnu vrijednost,
- implementirati metod `find(value)` koji pretražuje kolekciju i kao rezultat vraća iterator na pronađenu vrijednost. Ukoliko se vrijednost ne nalazi u kolekciji, metod vraća iterator na kraj kolekcije (`end()`),
- implementirati metod `remove(it)` koji briše element iz kolekcije sa pozicije na koju pokazuje iterator `it` te kao rezultat ne vraća nikakvu povratnu vrijednost,
- implementirati metode `find_if(predikat)` te `remove_if(predikat)` koji koriste predikat funkciju za detekciju članova,
- modificirati postojeći metod `invert` tako da implementacija metoda ne koristi dodatnu alokaciju memorije.

Iznad deklaracije svake od gore navedenih metoda, potrebno je ostaviti komentar sa složenosti tog metoda u O notaciji.

Primjer korištenja koda:

```
ArrayList<char> list1{'T', 'e', 's', 't'};
ArrayList<char>::const_iterator it = list1.cbegin() + 2;

list1.insert(it, 'x');
auto it2 = list1.find('s');
auto it3 = list1.find_if([](char c){return c > 'a' && c < 'm'});

list1.remove(it2);
list1.remove_if([](char c){return c == 't'});

list1.invert();
```

Zadatak 2

Napisati program koji bi poslužio biolozima za pohranu DNA lanaca. DNA lanac je sačinjen kao niz A, T, C i G molekula. Dužina ovog niza je proizvoljna i biolog ne zna puni izgled niza, već pomoću ovog programa unosi dio po dio DNA lanca, kako njegova istraživanja napreduju.

Kada se starta, program treba da ispiše meni za korisnika. Primjer korištenja programa je dat u nastavku:

Welcome to DNA storage. Please enter one of the following options:

1. Print
2. Print <pos> <len>
3. Insert <pos> <lanac>
4. Remove <pos> <len>
5. Store <file>
6. Load <file>

Your choice:

Podržane operacije su:

- print
Ispisuje sve poznate članove lanca. Pogledati primjer ispisa ispod.
- print <pos> <len>
Ispisuje sa pozicije <pos> najviše <len> članova lanca koji su poznati. Ukoliko dođe do dijela lanca koji nije poznat, ispisuje manje od <len> članova.
- insert <pos> <lanac>
Ubacuje u lanac na poziciju <pos> članove koji su navedeni u <lanac>. Pri tome prepisuje eventualne podatke koji su od ranije uneseni u pohranu.
- remove <pos> <len>
Briše najviše <len> elemenata iz lanca sa pozicije <pos>. Ukoliko dođe do dijela lanca koji nije poznat, briše manje od <len> članova. Ukoliko je lanac duži od <len>, ostatak ostaje u lancu.
- store <file>
Pohranjuje sve poznate članove u datoteku sa imenom <file>. Format datoteke je dat u nastavku.
- load <file>
Učitava podatke o lancu iz datoteke <lanac> pri čemu prebriše svaki stari unos. Pretpostavlja se da je datoteka uvijek pravilno formatirana.

Primjer rada (boldirani tekst predstavlja unos korisnika):

Your choice: 0

0-0: NULL

Your choice: 3

Position: 0

Value: **AACTCCA**

Your choice: 0

0-6: AACTCCA

Your choice: 2
Position: 0
Length: 3
0-2: AAC

Your choice: 2
Position: 0
Length: 100
0-6: AACTCCA

Your choice: 3
Position: 40
Value: **ATGGGGTA**

Your choice: 0
0-6: AACTCCA
7-39: NULL
40-47: ATGGGGTA

Your choice: 2
Position: 0
Length: 100
0-6: AACTCCA

Your choice: 4
Position: 0
Length: 100

Your choice: 0
0-39: NULL
40-47: ATGGGGTA

Your choice: 5
File: **data.txt**

Your choice: 6
File: **data.txt**

Datoteka koja služi za pohranu je organizirana u redove, gdje jedan red predstavlja jednu neprekinutu sekvencu u lancu. Red počinje sa pozicijom na kojoj počinje sekvenca, potom prazno mjesto, pa članovi sekvence.

Konkretno, za primjer iznad

0-6: AACTCCA
7-39: NULL
40-47: ATGGGGTA

Datoteka za pohranu bi izgledala ovako:

```
0  AACTCCA
40 ATGGGGTA
```

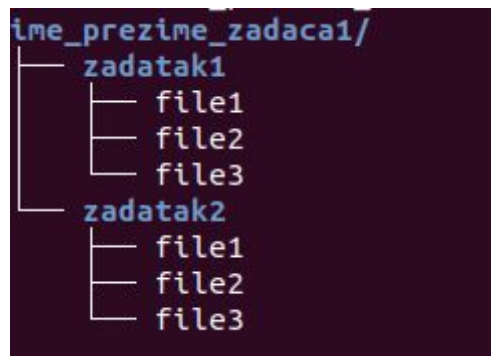
Prilikom implementacije dozvoljeno je koristiti strukture podataka i algoritme iz standardne biblioteke koje smo do sada radili na vježbama.

U prilogu zadaće se nalazi header file `dna_storage.hpp`. Studenti su dužni poštivati interface koji je dat u tom fileu. U tijelo klase studenti smiju dodati nove, privatne metode i članove klase kako smatraju potrebnim, međutim potpisi public metoda se ne smiju promijeniti. Implementaciju ovih metoda dodati u datoteku sa imenom `dna_storage.cpp`. Organizaciju ostatka koda, pomoćne funkcije i main funkciju, kao i dodatne *.cpp fileove studenti organizuju po potrebi.

Dodatno u prilogu zadaće je i cpp file sa primjerom rada sa fileovima u c++ programskom jeziku.

Način predavanja

Zadaću je potrebno predati u obliku arhive **ime_prezime_zadaca1.zip** pri čemu sadržaj arhive treba da bude sljedeći (nakon otpakivanja arhive na disku mora biti sljedeći sadržaj):



```
ime_prezime_zadaca1/
├── zadatak1
│   ├── file1
│   ├── file2
│   └── file3
└── zadatak2
    ├── file1
    ├── file2
    └── file3
```

Pored toga, potrebno je da se studenti strogo pridržavaju naziva metoda, njihovih parametara i povratnih vrijednosti.

Studenti koji se ne budu strogo pridržavali uputa neće biti adekvatno ocijenjeni. Svaki pokušaj prepisivanja će biti prijavljen i adekvatno sankcionisan.