ELEKTROTEHNIČKI FAKULTET
BANJA LUKA

PROGRAMSKI JEZICI I - 2. kolokvijum (14.12.2010) – A

❶

Definisati klasu **Par** koja omogućava manipulaciju jednim parom riječi. Svaki par riječi čine dva stringa sa proizvoljnim ("neograničenim") brojem znakova.

Klasa **Par** treba da ima:

a) **podrazumijevani konstruktor** koji kreira prazan par riječi;
b) **konstruktor** koji prima dva stringa i inicijalizuje par riječi;
c) odgovarajući **konstruktor kopije**;
d) odgovarajući **destruktor**.

Za klasu **Par** treba preklopiti sljedeće **operatore**:

a) **operator>>** - sa odgovarajućeg ulaza učitava jedan par riječi. Omogućiti kaskadno pozivanje.
b) **operator<<** - na zadanom izlazu ispisuje par riječi u obliku: `(prvarijec – drugarijec)`. Omogućiti kaskadno pozivanje.
c) **operator=** - jednom objektu klase **Par** dodjeljuje drugi objekat iste klase, pri čemu onemogućava samododjeljivanje (tj. onemogućava dodjelu `x=x`).
d) **operator<** - vraća rezultat poređenja dva para riječi na osnovu poređenja prve riječi u oba para, tj. ako je prva riječ u prvom paru "manja" od prve riječi (po abecedi) u drugom paru vraća istinu, inače vraća laž.
e) **operator[]** - kao rezultat vraća odgovarajuću riječ na osnovu zadate pozicije u paru (ako je argument jednak 1, funkcija vraća prvu riječ, a ako je argument jednak dva, funkcija vraća drugu riječ iz para).

Napomene:

1. Razdvojiti interfejs od implementacije, tj. funkcije članice definisati izvan definicije klase.
2. Dozvoljeno je korišćenje zaglavlja `<cstring>`, odnosno funkcija `strlen` i `strcpy`.

❷

Definisati klasu **Rjecnik** koja omogućava manipulaciju rječnikom. Osim neograničenog niza kojeg čine parovi riječi, rječnik ima i brojač koji sadrži trenutni broj riječi (trenutni broj parova) u rječniku.

Klasa **Rječnik** treba da ima:

a) **konstruktor** koji kreira prazan rječnik,
b) odgovarajući **destruktor**,
c) prijateljsku funkciju **`getBrojRijeci()`** koja vraća trenutni broj parova riječi u rječniku.

Za klasu **Rjecnik** treba preklopiti sljedeće operatore:

a) **operator<<** - na zadatom izlazu ispisuje sve parove riječi koje se trenutno nalaze u rječniku po abecednom redu, kao što je ilustrovano u primjeru, pri čemu treba omogućiti kaskadno pozivanje.
b) **operator+=** - omogućava da se u rječnik doda novi par riječi (rjecnik += noviParRijeci).
c) **operator()** - vraća prevod odgovarajuće riječi, odnosno prazan string ako data riječ ne postoji u rječniku.

Napomena:

1. Razdvojiti interfejs od implementacije, tj. funkcije članice definisati izvan definicije klase.

❸

Napisati program u kojem treba:

a) formirati prazan srpsko-engleski rječnik, a zatim sa standardnog ulaza učitati *n* (*n*>0) odgovarajućih parova riječi (prva riječ na srpskom, a druga njen prevod na engleskom) i napuniti rječnik.
b) na standardnom izlazu ispisati rječnik.
c) sa standardnog ulaza učitati neku riječ na srpskom jeziku, pa u rječniku pronaći njen prevod na engleski jezik i ispisati prevod na standardnom izlazu, kao što je ilustrovano u primjeru.
d) sa standardnog ulaza učitati neku riječ na engleskom jeziku, pa u rječniku pronaći njen prevod na srpski jezik i ispisati prevod na standardnom izlazu, kao što je ilustrovano u primjeru.

Primjer izvršavanja programa

```
n=3
   1. par: program program
   2. par: programski programming
   3. par: jezik language

Sadrzaj srpsko-engleskog rjecnika:
  slovo J
  ---------------------------------
  (jezik – language)
  =================================
  slovo P
  ---------------------------------
  (program – program)
  (programski – programming)
  =================================

Unesite rijec na srpskom: jezik
Njen prevod: language

Unesite rijec na engleskom: ball
Ne postoji ta rijec u rjecniku.
```

Raspodjela bodova po zadacima

Zadatak	1	2	3	Σ
Bodovi	50%	35%	15%	100%

ELEKTROTEHNIČKI FAKULTET
BANJA LUKA

PROGRAMSKI JEZICI I - 2. kolokvijum (14.12.2010) – B

❶

Definisati klasu **Dvojka** koja omogućava manipulaciju jednim uređenim parom. Prvi element para je cjelobrojni podatak, a drugi element para je string sa proizvoljnim ("neograničenim") brojem znakova.

Klasa **Dvojka** treba da ima:

a) **podrazumijevani konstruktor** koji kreira praznu dvojku, čiji je cjelobrojni podatak jednak nuli, a string prazan;
b) **konstruktor** koji prima jedan cijeli broj i jedan string pa inicijalizuje dvojku;
c) odgovarajući **konstruktor kopije**;
d) odgovarajući **destruktor**.

Za klasu **Dvojka** treba preklopiti sljedeće **operatore**:

a) **operator>>** - sa odgovarajućeg ulaza učitava jednu dvojku u obliku: `broj string`. Omogućiti kaskadno pozivanje.
b) **operator<<** - na zadatom izlazu ispisuje dvojku u obliku: `(broj,string)`. Omogućiti kaskadno pozivanje.
c) **operator=** - jednom objektu klase **Dvojka** dodjeljuje drugi objekat iste klase, pri čemu onemogućava samododjeljivanje (tj. onemogućava dodjelu `x=x`).
d) **operator==** - vraća rezultat poređenja dvije dvojke. Za dvije dvojke možemo da kažemo da su jednake ako su im odgovarajuće cjelobrojne komponente jednake. Ako su dvojke jednake vraća istinu, inače vraća laž.
e) **operator()** - kao rezultat vraća string sadržan u datom uređenom paru.

Napomene:

1. Razdvojiti interfejs od implementacije, tj. funkcije članice definisati izvan definicije klase.
2. Dozvoljeno je korišćenje zaglavlja `<cstring>`, odnosno funkcija `strlen` i `strcpy`.

❷

Definisati klasu **Mapa** koja omogućava manipulaciju jednostavnom kodnom mapom. Svaka mapa sadrži neograničen niz dvojki. Osim toga, mapa ima još jedan podatak član koji sadrži trenutni broj dvojki u mapi.

Klasa **Mapa** treba da ima:

a) **konstruktor** koji kreira praznu mapu,
b) odgovarajući **destruktor**,
c) prijateljsku funkciju **`getKapacitet()`** koja vraća trenutni broj dvojki u mapi.

Za klasu **Mapa** treba preklopiti sljedeće operatore:

a) **operator<<** - na zadatom izlazu ispisuje sve dvojke koje se trenutno nalaze u mapi, pri čemu treba omogućiti kaskadno pozivanje.
b) **operator+=** - omogućava da se u mapu doda nova dvojka (mapa += novaDvojka). Ako u mapi već postoji ista dvojka (tj. dvojka sa istom cjelobrojnom komponentom), ne treba dodati novu dvojku, već postojeću zamijeniti novom.
c) **operator()** - vraća string iz odgovarajuće dvojke u mapi, čija je cjelobrojna komponenta jednaka argumentu funkcije, odnosno prazan string ako u mapi ne postoji dvojka sa tim cjelobrojnim kodom.

Napomena:

1. Razdvojiti interfejs od implementacije, tj. funkcije članice definisati izvan definicije klase.

Primjer izvršavanja programa

```
n=4
   1. dvojka: 2 dva
   2. dvojka: 12 dva
   3. dvojka: 13 trinaest
   4. dvojka: 12 dvanaest

Sadrzaj mape:
  (2,dva)
  (12,dvanaest)
  (13,trinaest)

Unesite broj: 13
Odgovarajuci string: trinaest

Unesite broj: 2
Odgovarajuci string: dva

Unesite broj: 100
Ne postoji odgovarajuci string u mapi!
KRAJ!
```

❸

Napisati program u kojem treba:

a) formirati praznu mapu, a zatim sa standardnog ulaza učitati n (n>0) dvojki i napuniti mapu.
b) na standardnom izlazu ispisati mapu.
c) sa standardnog ulaza učitavati cijele brojeve i ispisivati odgovarajuće stringove iz mape, sve dok u mapi postoji dvojka čija je cjelobrojna komponenta jednaka učitanom cijelom broju, kao što je ilustrovano u primjeru.

Raspodjela bodova po zadacima

Zadatak	1	2	3	Σ
Bodovi	50%	35%	15%	**100%**