AS-0.101 C/C++ -ohjelmoinnin peruskurssi / Tentti 07.01.2003 / Hannu Laine

Vastaa neljään kysymykseen! Tentin arvosteluasteikko on 0 – 24 pistettä. Kaikkien kysymysten painoarvo on sama (6 pistettä/tehtävä). **Lähtökohtana on, että kysymyksessä 2 käytetään C++-kieltä ja muissa C:tä.** Jos kuitenkin kysymyksessä 2 matriisi on toteutettu oikeaoppisesti abstraktina tietotyyppinä C:llä, voi ko. tehtävästä saada 4 pistettä.

1.

Tietty määrä henkilöitä täyttää kysymyslomakkeen, jossa on 10 kysymystä. Jokaiseen kysymykseen vastataan antamalla vaihtoehdoksi luku 1, 2, 3, 4 tai 5. **Laadi ohjelma**, joka ensin kysyy vastanneiden henkilöiden lukumäärän. Tämän jälkeen se lukee annetut vastaukset siten, että ensin luetaan ensimmäisen henkilön vastaukset lähtien kysymyksestä 1 ja päätyen kysymykseen 10, sitten toisen henkilön antamat vastaukset samalla tavalla jne. Kun kaikkien henkilöiden vastaukset on syötetty, ohjelma selvittää mihin kysymyksiin kaikki ovat vastanneet samalla tavalla. Lopuksi ohjelma tulostaa niiden kysymysten numerot, joihin kaikki ovat vastanneet samalla tavalla.

Huomautus 1. Tehtävä pitää funktioilla jakaa tehtäväkuvauksen mukaisiin tai muuten järkeviin osiin ja tietojen välitys on tehtävä parametreilla. Globaaleja muuttujia ei saa käyttää.

Huomautus 2. Syöttötiedoille ei tarvitse tehdä muuta järkevyystarkastusta kuin, että vastausnumero on välillä 1…5.

Huomautus 3. Ohjelman käyttäjälle ei tarvitse antaa mitään ohjeita tai kehotteita.

2.

Ohjelmistoprojektissa käsitellään floating point lukujen matriiseja. Toteuta tämän projektin käyttöön C++:lla luokka Matrix (tai vaihtoehtoisesti C:llä abstraktina tietotyyppinä sama asia). Matriiseille tarvitaan seuraavat operaatiot:

konstruktori
areAdditive
add
areMultipliable
mul
getDim

Konstruktorille annetaan parametrina matriisin rivien määrä ja sarakkeiden määrä. Konstruktori varaa tilan matriisin alkioille ja alustaa alkiot nolliksi. Jäsenfunktiolla areAdditive, voidaan selvittää ovatko kaksi matriisia yhteenlaskettavia (eli onko niillä samat dimensiot).Jäsenfunktio add laskee yhteen kaksi matriisia. Jäsenfunktiolla areMultipliable, voidaan selvittää ovatko kaksi matriisia kerrottavissa keskenään (onko vasemman puolen

matriisin sarakkeiden määrä sama kuin oikean puolen matriisin rivien määrä). Jäsenfunktio mul tekee kahdelle matriisille kertolaskun. Jäsenfunktiolla getDim voidaan selvittää matriisin dimensiot (rivien määrä ja sarakkeiden määrä).

Kirjoita luokan Matrix luokkamäärittely ja toteuta sen yllämainitut jäsenfunktiot.

Huomautus 1. Jäsenfunktiot add ja mul saa toteuttaa myös operaattoreina + ja *.

3.

Sovelluksessa tarvitaan paljon funktioita, joilla kaikilla on prototyyppi muotoa int f(int a, int b); Funktiot (niiden osoitteet) tallennetaan taulukkoon, josta niitä on helppo hakea indeksillä (eli niiden paikkanumerolla taulukossa). Funktiotaulukon luonti, yksittäisen funktion tallentaminen taulukkoon ja funktion haku taulukosta halutaan tehdä selkeillä funktioilla, joiden nimet ovat CreateFuncArray, SetFuncToArray ja GetFuncFromArray.

Funktiolle CreateFuncArray annetaan parametriksi funktioiden maksimimäärä. Se varaa tilan funktiotaulukolle ja palauttaa funktiotaulukon osoitteen.

Funktiolle SetFuncToArray annetaan parametreina funktiolta CreateFuncArray saatu funktiotaulukko, funktio joka tallennetaan funktiotaulukkoon ja paikan indeksi taulukossa, joka ilmoittaa mihin kohtaan taulukossa funktio tallennetaan. Funktio SetFuncToArray yksinkertaisesti vain tallentaa funktion funktiotaulukkoon.

Funktiolle GetFuncFromArray annetaan parametreina funktiotaulukko ja paikan indeksi taulukossa, josta funktio haetaan. Funktio GetFuncFromArray palauttaa funktion funktiotaulukosta parametrin ilmoittamasta paikasta.

Kirjoita yllämainitut kolme funktiota ja lyhyt pääohjelma, jolla näytät kuinka funktioita käytetään. Pääohjelmassa luodaan kahden funktion taulukko siten, että alkioon 0 tallennetaan funktio int sum(int a, int b) ja alkioon 1 funktio int mul(int a, int b). Sitten pääohjelmassa haetaan funktio taulukon alkiosta 1 ja kutsutaan sitä parametreilla 10 ja 20, jolloin siis tulee suoritetuksi lukujen 10 ja 20 kertolasku, koska oletetaan, että funktio mul suorittaa kertolaskun.

Huomautus 1. Funktioita sum ja mul ei tarvitse kirjoittaa, koska ne ovat itsestään selviä.

4.

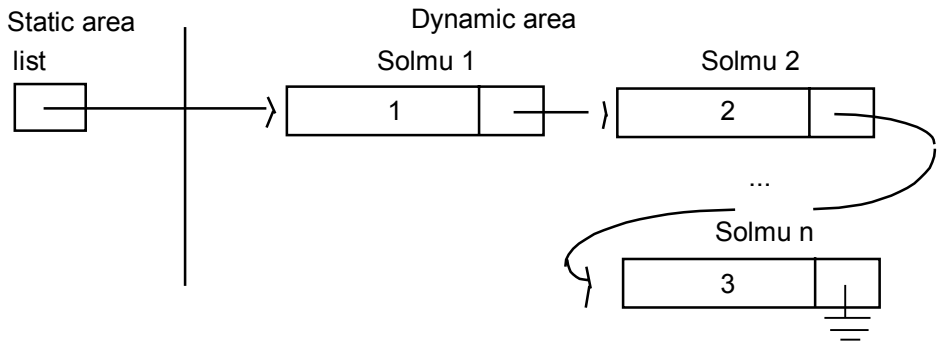
Kokonaislukujen listalle tehdään dynaamisesti linkattu toteutus siten, että koko listaa edustaa yksi osoitin ensimmäiseen dynaamisella alueella olevaan solmuun (katso kuva alla). **Kirjoita tarvittavat tietomäärittelyt** tätä listaa edustavalle tyypille Tlist. **Kirjoita sille lisäksi seuraavat kolme operaatiofunktiota:**

initialize, joka alustaa listaa edustavan osoittimen siten, että se sisältää NULL-osoittimen, joka edustaa tyhjää listaa.

insert_nth_node, joka lisää uuden solmun linkattuun listaan siten, että se tulee n:nneksi solmuksi listassa. Järjestysnumero annetaan parametrina (järjestysnumero nolla tarkoittaa ensimmäistä alkiota jne). Myös uuden solmun tietosisältö (kokonaisluku) annetaan parametrina.

combine_lists, joka muodostaa annetusta kahdesta listasta (seuraavassa lista 1 ja lista 2) yhden siten, että siinä on ensimmäisenä solmuna listan 1 ensimmäinen solmu, toisena solmuna listan 2 ensimmäinen solmu, kolmantena solmuna listan 1 toinen solmu, neljäntenä solmuna listan 2 toinen solmu jne. Jos toinen lista on toista pitempi, pitemmän listan ylimääräiset solmut jäävät alkuperäiseen listaan. Yhdistäminen pitää tehdä siten, että vain osoittimia modifioidaan.. Uusia solmuja ei saa luoda muistiin eikä solmun datasisältöjä (kokonaislukuja) saa siirtää solmusta toiseen.

Huomautus 1. Erikoistapaukset pitää tietysti hoitaa asianmukaisesti (tyhjä lista, vain yksi alkio listassa tai lisätään ensimmäiseksi, keskelle, viimeiseksi jne). NULL-osoitin edustaa tyhjää listaa.



5.

Vertaile funktioiden ja parametrillisten makrojen ominaisuuksia.