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 lukujen joukko on toteutettu oikeaoppisesti abstraktina tietotyyppinä C:llä, voi ko. tehtävästä saada 4 pistettä.

1.

Eräässä ohjelmassa, jota käytetään kielentutkimukseen, lause on tallennettu yhteen merkkitaulukkoon. Lauseen sanat on erotettu taulukossa välilyönnillä, kuten tekstissä yleensäkin. Ohjelmassa halutaan saada tehokas pääsy yksittäisiin sanoihin ja siksi lauseen sanoja halutaan käsitellä erillisinä C:n merkkijonoina. **Kirjoita tätä varten funktio split**, joka saa parametrina alkuperäisen lauseen sisältävän merkkijonon (siis merkkitaulukon) ja tuottaa tuloksena lauseessa olevien sanojen määrän ja osoitintaulukon, joka sisältää osoittimet lauseen sanoihin. Silloinhan tämän funktion tuottamaa osoitintaulukkoa indeksoimalla päästään suoraan käsiksi yksittäisiin sanoihin. Jos t edustaisi tätä funktion tuottamaa osoitintaulukkoa, niin voisimme tulostaa esimerkiksi neljännen sanan lauseesta ilmaisulla printf("%s", t[3]);

Huomautus. Lauseessa mahdollisesti olevia muita välimerkkejä kuin välilyönti ei tarvitse huomioida.

2.

Tietoalkioiden joukko (set) määritellään seuraavasti. Joukko on tietoalkioiden kokoelma, jossa alkioiden kesken ei ole määritelty mitään järjestystä. Lisäksi joukko voi sisältää vain yhden esiintymän tiettyä alkiota. Tämän takia joukon tila ei muutu, kun sinne lisätään alkio, joka jo on kerran lisätty. Tässä tehtävässä toteutetaan kokonaislukujen joukko C++:n luokkana (tai vaihtoehtoisesti abstraktina tietotyyppinä C:llä). Kokonaislukujoukon luokkanimi on IntSet ja sillä on seuraavat jäsenfunktiot:

IntSet (konstruktori)
addItem
removeItem
isItemInSet
numberOfItems
capacity

Konstruktori alustaa joukon käyttökuntoon. Konstruktorille annetaan parametrina joukon maksimikoko. Jäsenfunktiolla addItem voidaan lisätä yksi parametrina annettu kokonaisluku joukkoon. Jäsenfunktiolla removeItem voidaan poistaa parametrina annettu luku joukosta. Jäsenfunktiolla isItemInSet voidaan selvittää, onko parametrina annettu luku joukossa. Funktio palauttaa arvon 1, jos parametrina annettu luku on joukossa. Muuten se palauttaa arvon 0. Jäsenfunktiolla numberOfItems voidaan selvittää, montako alkiota säiliössä parhaillaan on

Jäsenfunktiolla capacity voidaan selvittää, mikä on joukon maksimikoko eli montako alkiota sinne korkeintaan mahtuu.

Kirjoita luokan IntSet esittely ja sen jäsenfunktioiden toteutukset.

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.

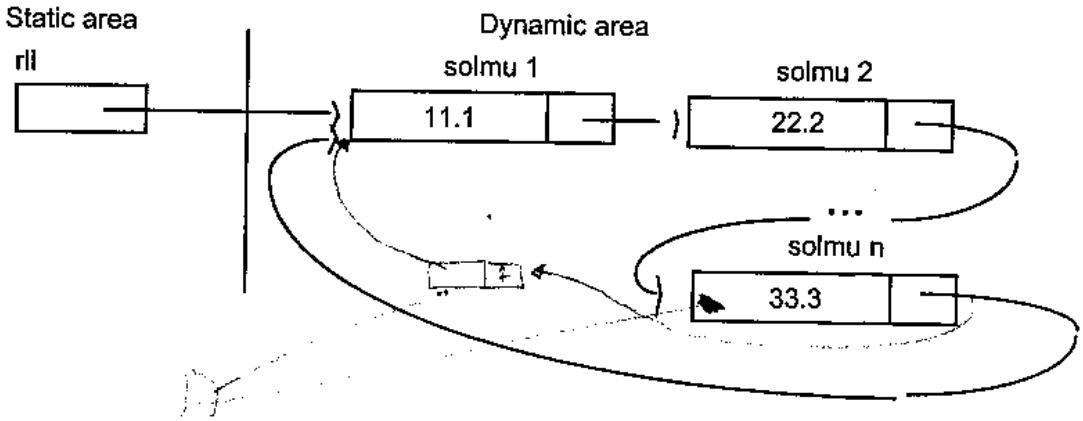
Funktiolle GetFuncFromArray annetaan parametreina funktiotaulukko ja paikan indeksi taulukossa, josta funktio haetaan. Funktio GetFuncFromArray palauttaa funktion funktiotaulukosta parametrin ilmoittamasta paikasta.

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.

Kirjoita tietomäärittelyt, jotka tarvitaan alla kuvatun renkaaksi linkitetyn listan (rll) kuvaamiseen. Renkaaksi linkitetyn listan solmuissa (node) tietokenttä (data field, item field) on floating point luku. Huomaa, että yksi osoitin edustaa rengasta. **Kirjoita funktio laske_solmut**, joka laskee ja palauttaa tiedon, kuinka monta solmua parametrina annettu renkaaksi linkitetty lista sisältää. **Kirjoita lisäksi funktio lisaa_solmu**, joka lisää parametrina annettuun renkaaseen yhden solmun siten, että renkaan solmuosoitin osoittaa lisäyksen jälkeen uuteen solmuun. Uuden solmun tietosisältö annetaan funktiolle parametrina.



Huom 1. Erikoistapaukset pitää tietysti hoitaa asianmukaisesti (tyhjä lista, vain yksi alkio listassa jne). NULL-osoitin edustaa tyhjää listaa.

5.

Vertaile dynaamisia muuttujia ja staattista muuttujia. Vertailussa tarkastellaan ominaisuuksia, käyttöä, mahdollisuuksia, rajoituksia jne. Staattisina muuttujina pidetään kaikkia muuttujia, jotka eivät ole dynaamisessa muistissa (kasassa, heapissä).