AS-0.101 C/C++ -ohjelmoinnin peruskurssi / Tentti 10.12.2003 / 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 merkkikassi on toteutettu oikeaoppisesti abstraktina tietotyyppinä C:llä, voi ko. Tehtävästä saada 4 pistettä.

1.

Huoneessa tehdään lämpötilamittauksia viidessä eri pisteessä viikon aikana eli seitsemänä päivänä. **Kirjoita ohjelma**, jolle syötetään ensin mittauspisteessä 1 mitatut seitsemän arvoa, sitten mittauspisteessä 2 mitatut seitsemän arvoa jne. Kun kaikki mitatut arvot on syötetty, ohjelma, selvittää minä päivänä huoneen keskilämpötila (eri pisteissä mitattujen lämpötilojen keskiarvo) oli suurin. Riittää kun ohjelma tulostaa näyttöön viikonpäivän numeron (1, 2, ... tai 7).

Huomautus 1. Tehtävä pitää funktioilla jakaa sopivasti osiin ja tietojen välitys on tehtävä parametreillä. Globaaleja muuttuj ia ei saa käyttää.

Huomautus 2. Syöttötiedoille ei tarvitse tehdä mitään järkevyystarkastuksia.

2.

Joukko on tietojen säiliö, jossa yksi tieto voi esiintyä vain kerran. Kassi taas on tietojen kokoelma, johon voidaan tallentaa sama tieto useampia kertoja ja kassi on tietoinen myös kunkin sinne tallennetun tiedon esiintymiskerrasta. Kassi voidaan toteuttaa luokkana (tai abstraktina tietotyyppinä). Tässä tehtävässä toteutetaan merkkien kassi (luokka TcharBag), jonne voidaan siis tallentaa yksittäisiä merkkejä. Merkkikassilla on seuraavat jäsenfunktiot

TcharBag (konstruktori)
add
remove
getCount

Konstruktorilla tehdään kassi käyttövalmiiksi ja sen sisältö tyhjäksi. Jäsenfunktiolla add voidaan lisätä yksi parametrinä annettu merkki kassiin. Jäsenfunktiolla remove voidaan poistaa parametrinä annetun merkin yksi esiintymä kassista ja jäsenfunktiolla getCount voidaan kysyä parametrinä annetun merkin esiintymiskertojen lukumäärä kassissa.

Kirjoita luokan TcharBag esittely ja sen jäsenfunktioiden toteutukset.

3.

C-kielen standardikirjaston merkkijonojen käsittelyfunktioissa on yleensä lähtökohtana se, että tila mahdolliselle tulosmerkkijonolle on valmiiksi varattuna. Esimerkiksi funktion strcpy (merkkijonon kopiointifunktio), jonka prototyyppi on

char * strcpy(char *destination, const char * source);

tapauksessa parametrin destination osoittamalla muistialueella tulee olla riittävästi tilaa merkkijonon source kopiolle. C-kielen standardikirjastosta myös puuttuu joitakin käteviä mm. Basic-kielestä tuttuja funktioita kuten osan ottaminen merkkijonosta lähtien tietystä merkkipaikasta ottamalla tietty määrä merkkejä siitä eteenpäin.

Kirjoita C-kielen merkkijonoille seuraavat kolme funktiota:

substring
allocstrcpy
allocstrcat

Funktiolle substring välitetään kolme parametriä: 1) merkkijono, josta otetaan osa, 2) sen merkkipaikan indeksi (numero), josta lähtien alastringi otetaan ja 3) alastringin pituus eli otettavien merkkien määrä. Funktio palauttaa osoittimen alastringiin. Funktio varaa alastringille juuri sopivan kokoisen tilan dynaamisesta muistista. Alkuperäinen stringi säilyy siis muuttumattomana.

Funktiolla allocstrcpy voidaan tehdä kopio merkkijonosta. Ero standardikirjastossa olevaan funktioon strcpy on siinä, että allocstrcpy varaa tilan kopiolle dynaamisesta muistista (juuri sopivan kokoisena). Funktiolla pitää olla edelleen kaksi parametriä, joista ensimmäisellä ilmaistaan kohdetta (destination) ja toisella lähdettä (source).

Funktio allocstrcat liittää toisen parametrin edustaman merkkijonon ensimmäisen parametrin edustaman merkkijonon perään. Tässäkin tapauksessa lähtökohta on se, että ei voida olettaa, että sen merkkijonon perässä, johon liitetään olisi tilaa ylimääräisille merkeille. Siksi funktio allocstrcat varaa juuri sopivan pituisen muistialueen tulosstringille. Samoin kuin standardifunktion strcat tapauksessa, merkkijono, jonka perään liitetään ei enää ole sellaisenaan erillisenä olemassa funktion käytön jälkeen.

Näyttääksesi kuinka kirjoittamiasi funktioita käytetään, **laadi pieni pääohjelma**, joka ensin lukee näppäimistöltä kaksi merkkijonoa (string1 ja string2). Sitten ohjelma tekee kopion ensimmäisestä luetusta merkkijonosta (copyString) dynaamiselle alueelle funktiolla allocstrcpy. Seuraavaksi ohjelma muodostaa uuden merkkijonon (leftString) 4:stä ensimmäisestä merkkijonon string2 merkistä funktiolla substring. Lopuksi ohjelma muodostaa tulosmerkkijonon (resultString) liittämällä ensimmäisen luetun merkkijonon kopioon copyString toisen luetun merkkijonon alun leftString. Tällöin siis molemmat alunperin luetut merkkijonot pysyvät muistissa muuttumattomina.

Huomautus. Standardifunktion strcat prototyyppi on
char * strcat(char *destination, const char * source);

4.

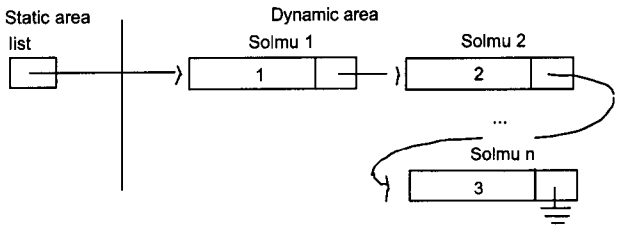
Kokonaislukujen listalle tehdään dynaamisesti linkattu toteutus siten, että koko listaa edustaa yksi osoitin ensimmäiseen dynaamisella alueella olevaan solmuun (katso kuva alla). **Kirjoita tarvittavat tietomäärittelyt** tätä listaa edustavalle tyypille Tlist. **Kirjoita sille lisäksi seuraavat kolme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

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

delete_nth_node, joka poistaa n:nnen solmun listasta. Poistettavan solmun järjestysnumero (alkaa nollasta) annetaan parametrina.

split_list, joka muodostaa annetusta listasta kaksi listaa siten, että ensimmäisessä on alkuperäisen listan järjestysnumeroltaan parilliset solmut (siis nollas, toinen, neljäs jne) ja toisessa alkuperäisen listan parittomat solmut. Splittaus 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 jne tai poistetaan ensimmäinen alkio, viimeinen tai mikä tahansa alkio). NULL-osoitin edustaa tyhjää listaa.



5.

Selitä **osoittimien ja taulukoiden yhteys** C-kielessä ja kuinka tätä yhteyttä hyödynnetään eri tilanteissa.