

Demo C1 / 23.4

Demoista C1 ja C2 on tehtävä yhteensä 60%, min 40%/kerta, jotta saa 1 op merkinnän kurssista *TIEP112 Ohjelmointi 2, C++*. Kurssi arvostellaan Hyväksytty/Hylätty. Demokertojen maksimina pidetään 10, eli käytännössä Hyväksyttyyn vaaditaan 12 tehtyä tehtävää.

Vähintään kahteen vastaukseen ”automaattinen” testaus perinteisellä tavalla.

1. Miten esikäntäjä muuttaisi seuraavan tiedoston? Onko muodostuvassa koodissa virheitä? Mitä ohjelma tulostaisi (mahdollisten korjausten jälkeen)? (Mukaeltu koekysymys kevät-93)

```
#define pois return 0;
#include <stdio.h>
#define HALPA=HINTA;
#define auto "Mosse"
#define Tul "Terve"
#define kaa "tuloa"
#define TASTA int main(void){
#define TAHAN pois }
#define VAHAN
#define PP ;
#define VEROJA (1+40 prosenttia) PP
#define plus *
#define on =
#define HINTA=7;
#define lf "\n"
#define prosenttia / 100.0

TASTA
    int maksu on HALPA plus VAHAN VEROJA
    printf("auto on vanha, mutta halpa: %d!\n",maksu) PP
    printf(Tulkaa " kyytiin!" lf) PP
TAHAN
```

2. Kirjoita C-ohjelma (käyttäen `stdio.h:n` funktioita, ei `iostream.h:n`), joka kysyy huoneesta mitatut tiedot ja tulostaa sitten näiden perusteella huoneen pinta-alan ja tilavuuden. Toteuta ohjelma [monisteen 8.5 luvun](#) mukaisesti aliohjelmia käyttäen (parametrin välitys osoittimien avulla, vrt. [moniste 8.5.4](#)). Katso malliksi [matka_a2.c](#).
3. Tee edellisen tehtävän huone-ohjelmasta tietovirroin ja viitemuuttujien avulla toteutettu C++-ohjelma (vrt. [matka_ar.cpp](#), `iostream.h` ja [8.5.5](#)).
4. Näytä kuvan avulla (piirrä kuva kunkin sijoituksen jälkeen uudelleen) mitä ovat muuttujien arvot seuraavien sijoitusten jälkeen (kun muuttujat ovat sijoittuneet muistipaikkoihin kuten kuvassa). Piirrä kuvaan myös mihin osoitin `p` loogisesti aina osoittaa.

```
int a,b,c;
int *p;
int e;
/* 1 */    a = 19;
/* 2 */    p = &b;
```

```

/* 3 */ *p = a+2;
/* 4 */ p = p+1; /* Tekee käytännössä p=p+2 */
/* 5 */ *p = 7;
/* 6 */ p = p+2; /* Tekee käytännössä p=p+4 */
/* 7 */ *p = p-3; /* Käytännössä p-6 */
/* 8 */ **p = 199;

```

```

/* 1 */ /* 2 */ /* 3 */ /* 4 */ /* 5 */ /* 6 */ /* 7 */ /* 8 */
+-----+ +-----+ +-----+ +-----+ +-----+ +-----+ +-----+ +-----+
100 |      | a |      | |      | |      | |      | |      | |      | |      |
+-----+ +-----+ +-----+ +-----+ +-----+ +-----+ +-----+ +-----+
102 | ??  | b |      | |      | |      | |      | |      | |      | |      |
+-----+ +-----+ +-----+ +-----+ +-----+ +-----+ +-----+ +-----+
104 | ??  | c |      | |      | |      | |      | |      | |      | |      |
+-----+ +-----+ +-----+ +-----+ +-----+ +-----+ +-----+ +-----+
106 | ??  | p |      | |      | |      | |      | |      | |      | |      |
+-----+ +-----+ +-----+ +-----+ +-----+ +-----+ +-----+ +-----+
108 | ??  | e |      | |      | |      | |      | |      | |      | |      |
+-----+ +-----+ +-----+ +-----+ +-----+ +-----+ +-----+ +-----+

```

Osoitinaritmetiikassa $p=p+1$ tarkoitetaan että osoitin p siirtyy seuraavaan "olioon". Siksi käytännössä 16-bittisessä koneessa muistipaikan arvo kasvaakin kahdella.

5. Tutki pöytätestin avulla mitä ovat muuttujien arvot seuraavassa ohjelmassa kunkin lauseen suorittamisen jälkeen. Mitä ohjelma tulostaa? (Lause $a=i++$; vastaa samaa kuin lauseet $a=i$; $i=i+1$; Lause $a=++i$; vastaa samaa kuin lauseet $i=i+1$; $a=i$;)

```

#include <stdio.h>
/* 01 */
/* 02 */ int *d,c=4,u=9;
/* 03 */
/* 04 */ int huijaus(int a,int *b)
/* 05 */ {
/* 06 */     *(d+1) = ++(*b);
/* 07 */     c = *b - 2;
/* 08 */     *b += a/2;
/* 09 */     return a > 3;
/* 10 */ }
/* 11 */
/* 12 */ int puijaus(int **b)
/* 13 */ {
/* 14 */     int n;
/* 15 */     *b = &u;
/* 16 */     *d =175;
/* 17 */     n = --u;
/* 18 */     return 3 * (*b == d) + n;
/* 19 */ }
/* 20 */
/* 21 */ int main(void)
/* 22 */ {
/* 23 */     int k1 = 31, k2 = 45; d = &c-1;
/* 24 */     k1 = huijaus(k2+*(d+1),&c);
/* 25 */     k2 = puijaus(&d);
/* 26 */     printf("%4d ",*d);
/* 27 */     printf("%4d %4d %4d %4d\n",u,c,k2,k1);
/* 28 */     return 0;
/* 29 */ }

```

6. Kirjoita ja testaa C-funktio `kysy_vuosi`, joka kysyy käyttäjältä vuosiluvun ja palauttaa käyttäjä kirjoittaman vuoden parametrilistassa. Funktion paluuarvona (`return`-lauseessa) palautetaan 1 mikäli annettu vuosiluku ei ole tältä vuosisadalta tai syötössä on jotakin väärin. Muutoin palautetaan 0.

7. Tee aliohjelma `pienimman_paikka`, joka palauttaa kokonaislukutaulukon pienimmän alkion paikan (indeksin). Esim. monisteen [luvun 13.1.4](#) taulukosta:

```
i = pienimman_paikka(k_pituudet,12); /* => i=1 */
```

Tee edellistä aliohjelmaa **käyttäen** aliohjelma `pienin`, joka palauttaa kokonaislukutaulukon pienimmän alkion arvon

```
n = pienin(k_pituudet,12); /* => n = 28 */
```

8. Kirjoita funktio `onko_palindromi`. Tee kaksi eri versiota. Toinen C-merkkijonoille ja toinen C++ -merkkijonoille (`string`-luokka).

9. Kirjoita kaksi eri aliohjelmaa (toinen `string`-jonoille ja toinen `char *`-jonoille) `tuhoa_lopusta`, jotka loogisessa mielessä "poistavat" merkkijonon `n` viimeistä merkkiä (muista virhetilanteet!):

```
char s[]="Kissa istuu";
tuhoa_lopusta(s,3); /* => s = "Kissa is" */
string st("Kissa istuu");
tuhoa_lopusta(st,3); // => st= "Kissa is"
```

10. Tee aliohjelma

```
int poista(int taulukko[],int lkm,int n)
```

joka poistaa taulukosta kaikki luvun `n` esiintymät.

```
int t[]={4,7,6,3,6,2};
int lkm=6;
```

```
lkm = poista(t,lkm,6); /* => t = {4,7,3,2}, lkm = 4 */
```