Tentamen Programmeermethoden
Donderdag 16 januari 2025
13:00–16:00 uur Informatica

Universiteit Leiden
The Netherlands

Bij alle functies moeten de variabelen (constanten eventueel uitgezonderd) in de heading of locaal voorkomen; vul zelf headings goed in. De te behalen punten (totaal 100) staan tussen haakjes bij de opgaven. Succes! Cijfers: te zijner tijd via Brightspace/uSis.

1. (25 punten) In het array `int A[n]` staan `n` (een `const int` ≥ 3) verschillende gehele getallen.
a. (5) Geef een C++-functie `negpos (A,n,pos,neg)` die in `pos` de som van alle positieve getallen oplevert, en in `neg` de som van alle negatieve.
b. (6) Schrijf een C++-functie `splits (A,n)` die ervoor zorgt dat alle negatieve getallen in `A` vooraan komen, en de positieve (≥ 0) achteraan. Begin met `i` vooraan en `j` achteraan, en laat deze naar elkaar toelopen.
c. (4) Geef een C++-functie `sorteer (A,n)` die `A` oplopend sorteert met *bubblesort*. Stop hierbij als er in een ronde geen verwisselingen waren.
d. (7) Schrijf een Booleaanse C++-functie `som3 (A,n)` die bepaalt of er drie *verschillende* getallen in `A` staan die samen tot 0 sommeren, door alle mogelijke drietallen te bekijken.
e. (3) Hoeveel drietallen array-elementen bekijkt het algoritme `som3` van **d** ongeveer in het slechtste geval, en wanneer is dat?

2. (25 punten) **a**. (6) Bij een functie kun je te maken hebben met *call by value* en *call by reference*, en ook met *locale* en *globale* variabelen. Verder onderscheiden we ook nog *formele* en *actuele* parameters. Leg deze zes begrippen duidelijk uit.
b. (7) Gegeven een C++-programma met daarin de volgende twee functies:

```
int bruce (int a, int b, int i) {
  while ( a < b ) {
    z = z + 2;
    if ( i % 2 == 0 ) { a++; } else { b--; }
    cout << "B " << a << "," << b << endl;
    i++; }//while
  return a; }//bruce
int peter (int u, int v) {
  int j, som = 0; z++;
  for ( j = 1; j <= 3; j++ ) {
    som += bruce (u,v,j);
    cout << "P " << j << "," << som << endl; }//for
  return som; cout << "2025" << endl; }//peter
```

Verder zijn de globale variabelen `x`, `y` en `z` van type `int` gegeven. Wat is dan de uitvoer van het volgende stukje programma (leg je antwoord duidelijk uit):

```
x = 3; y = 4; z = 35; cout << peter (x,y) << endl;
cout << x << "," << y << "," << z <<  endl;
```

c. (7) Als **b**, maar nu met een `&` bij alle 5 parameters van de functies.
d. (5) Wat is de return-waarde van `bruce` (zonder `&`), uitgedrukt in `a`, `b` en `i`?

3. (25 punten) Gegeven is een `m` bij `n` (beide `const int` > 0; ze hoeven bij deze opgave niet te worden doorgegeven als parameter) array `puzzel` met karakters. Een voorbeeld met `m` $= 4$ en `n` $= 5$ staat hiernaast. Hier heeft element `puzzel[2][3]` waarde `'4'`.

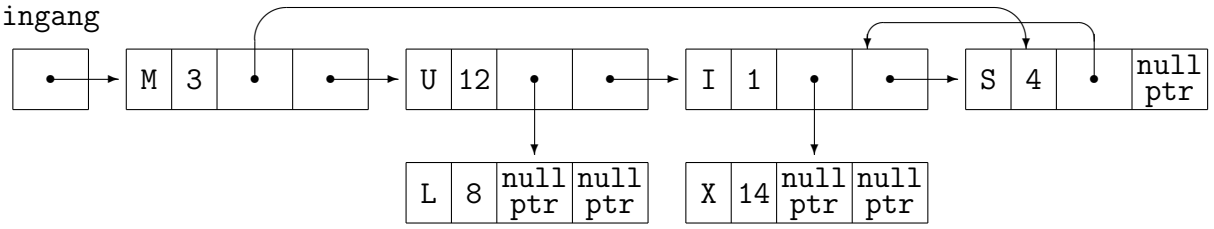
'1'	'k'	'S'	'2'	'3'
'h'	'a'	'p'	'J'	'e'
'a'	'r'	'e'	'4'	'l'
'L'	'5'	'6'	'7'	'k'

a. (8) Schrijf een C++-functie `int tellen (puzzel)` die telt hoevaak het in het array `puzzel` voorkomt dat twee dezelfde kleine letters direct diagonaal naast elkaar staan. In het voorbeeld 1 (de twee `a`'s).
b. (8) Schrijf een C++-functie `int gr (puzzel,j)` die de kolom `j` oplevert met de grootste totaalsom van voorkomende cijfers, en deze som retourneert. In het voorbeeld kolom 3, met som $2 + 4 + 7 = 13$. Als er meerdere kolommen dezelfde grootste som hebben, geef dan die met de kleinste index. De functie `isdigit` mag niet gebruikt worden.
c. (9) Schrijf een C++-functie `print (puzzel,c)` die het verticale *woord*, dat begint onder het vakje waarin cijfer `c` staat, afdrukt. Een *woord* bestaat uit niet-cijfers, en is zo lang mogelijk. Als voorbeeld `haL` voor cijfer `'1'`, `J` voor cijfer `'2'`, en niets voor cijfer `'4'` of `'5'`. Neem aan dat `c` hooguit één maal voorkomt in `puzzel`.

4. (25 punten) Gegeven is het volgende type:

```
class dier { public: char soort; int leeftijd; dier* vriend; dier* volg; };
```

Hiermee wordt een lijst met dieren (ieder van zekere `soort` en `leeftijd`) opgebouwd. Het veld `vriend` bevat een pointer naar een ander `dier`-object (soms in de lijst, soms niet (deze heeft dan zelf `nullptr` als `vriend` en als `volg`)), en `volg` wijst naar het volgende `dier`-object. Een voorbeeld (`ingang` van type `dier*`; `I` is de vriend van `S`, `L` die van `U`):



a. (6) Schrijf een C++-functie `voegtoe (ingang,s1,l1,s2,l2)` die een nieuw `dier`-object (soort `s1`, leeftijd `l1`) vooraan de lijst toevoegt, met een *nieuwe* vriend (soort `s2`, leeftijd `l2`; dus niet in de lijst).
b. (6) Schrijf een C++-functie `verwijder (ingang)` die het eerste `dier`-object uit de lijst (met `ingang` van type `dier*` als ingang) verwijdert, mits dat er is. Denk dus aan de lege lijst. Gooi ook een vriend die niet in de lijst staat weg (te herkennen aan de `nullptr`).
c. (3) Schrijf een C++-functie `wisselen (ingang)` die de soorten van de eerste en door diens `vriend`-pointer aangewezen `dier`-objecten verwisselt — indien deze bestaan en hun leeftijden samen oneven zijn. In het voorbeeld: `M` $\leftrightarrow$ `S`.
d. (3) In de functies bij **a**, **b** en **c** staat in de heading een pointer. Deze heb je call by value of call by reference doorgegeven (met een `&`). Maakt het voor de werking van deze functies verschil uit of die `&` erbij staat? Mag het, moet het? Leg duidelijk uit.
e. (7) Schrijf een C++-functie `int totaal (ingang,h1,h2)` die bepaalt hoeveel `dier`-objecten er zijn in de lijst met een vriend in de lijst (`h1`; hier 2) en met een vriend buiten de lijst (`h2`; hier 2). Geretourneerd moet worden de totaalsom van alle leeftijden (hier 42).