

---

C++ifPCwhileLinuxdiffintFirefoxboolpublicprivateWindows  
fileg++UNIXVisualOSFedoralscharpsopenclosegetputarray  
MaccpelsecoutmainRedHatmkdireditorstructDebianSublime  
classusingcmathChrometruecpSUSEnamespaceenuminclude  
doublecdGNUdokillobjectcompilermorelinkiostreamcinnotlpr  
WWWmakefilegrepiomanipgeditfloatGNOMEfstreamrmfalse  
andorCode::Blockstopforstringe-mailmanchmodWSLUbuntu

---

# Programmeermethoden



Universiteit  
Leiden

Walter Kusters en Jonathan Vis

week 1: 2–6 september 2024

Introductie

[www.liacs.leidenuniv.nl/~kusterswa/pm/](http://www.liacs.leidenuniv.nl/~kusterswa/pm/)

↑  
tilde (~)

E-mailadres: `pm@liacs.leidenuniv.nl`

Het college wordt gegeven door dr. W.A. (Walter) Kusters en dr. J.K. (Jonathan) Vis.



Werkcolleges worden verzorgd door studentassistenten:

**Donderdag:** Kay Gijzen, Leon Hoogendoorn, Imke van Ooijen, Valentijn Ouwehand en Miguel van der Wekken

**Vrijdag:** Zahir Bingen, Sam Testers en Daan van Vliet

Anderen geven PM-Python voor Bioinformatica, DSAI en Informatica & Economie; en voor Natuur/Sterrenkunde.

Er zijn verschillende soorten activiteiten:

**colleges** live in Gorlaeus 3; zie ook (oudere) video's:

[www.liacs.leidenuniv.nl/~kosterswa/pm/videos.php](http://www.liacs.leidenuniv.nl/~kosterswa/pm/videos.php)



**werkcolleges** in kleine (computer)zalen Gorlaeus; neem laptop mee; donderdagmiddag (DM.0.09/13/21) of vrijdagmiddag (BW.0.17/18/19; DM.0.13); er is een **aanwezigheidscontrole!**

**vragenuren** direct na werkcolleges; en op maandagmiddagen bij BM.2.07

Eerste week: 2–6 september 2024; laatste week: 9–13 december. Geen activiteiten op 3 en 4 oktober en in de week van 21–25 oktober.

[www.liacs.leidenuniv.nl/~kosterswa/pm/schema.php](http://www.liacs.leidenuniv.nl/~kosterswa/pm/schema.php)

| ma | di | wo | donderdag 5.9                                                                                         | vrijdag                                                                                               |
|----|----|----|-------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------|
|    |    |    | 11:00–12:45<br>Gorlaeus<br>zaal 3<br>college 1<br>iedereen                                            | 13:15–...<br>Gorlaeus<br>BW.0.17/18/19<br><a href="#">werkcollege 1</a> ,<br>vragenuur<br>Wiskundigen |
|    |    |    | 13:15–...<br>Gorlaeus<br>DM.0.09/13/21<br><a href="#">werkcollege 1</a> ,<br>vragenuur<br>Informatici |                                                                                                       |

Vragenuur: maandagmiddag 9 september in de buurt van  
Gorlaeus BM.2.07



“schotel”

Iedereen gebruikt zijn/haar **ULCN-account**. Daarmee kun je ook WiFi gebruiken in universiteitsgebouwen, via eduroam. Zet een “forward” voor e-mails!

Je krijgt daarmee ook een *home directory* (“homedir”) in het universitaire UNIX-systeem. In de computerzalen kun je daar via Linux mee werken.

Hoe kom je vanaf huis bij je UNIX-files? Antwoord: “ssh-en” en “scp-en” naar

`a.einstein@sshgw.leidenuniv.nl`

(en verder) als je `a.einstein` bent. Zie later.

Sluw: **laptop**, USB, eigen e-mail, de cloud, ...

Er moeten vier programmeeropgaven gemaakt worden. **Als** ze alle voldoende zijn (hooguit één 5) **en** het tentamen voldoende ( $\geq 5.5$ ) is gemaakt, krijg je zes studiepunten:

$$\text{Eindcijfer} = \frac{2 \times \text{Schriftelijk} + \frac{\text{Op}_1 + \text{Op}_2 + \text{Op}_3 + \text{Op}_4}{4}}{3}$$

(afgerond naar het dichtstbijzijnde element uit de verzameling  $\{1, 2, 3, 4, 5, 6, 6.5, 7, 7.5, 8, 8.5, 9, 9.5, 10\}$ ).

Heb je nog deelresultaten uit voorgaande jaren? Ga langs bij de docent!

Cijfers: [Brightspace](#) (meld je aan = “enroll”).

- in tweetallen maken
- **wel** overleggen en om hulp vragen, **niet** kopiëren;  
zie website voor huisregels en [richtlijnen](#), bijvoorbeeld  
voor aanvullen, ...
- [op tijd](#) inleveren: deadlines (**−1 per week te laat**)
- deels maken tijdens werkcolleges, deels thuis
- vragenuren!

- Opgave 1: maandag 23 september 2024, 18:00 uur
- Opgave 2: maandag 14 oktober 2024, 18:00 uur
- Opgave 3: maandag 11 november 2024, 18:00 uur
- Opgave 4: maandag 9 december 2024, 18:00 uur

Lever per team (“group”) de C++-code in Brightspace in.  
En het geprinte verslag in de doos bij kamer BM.2.07.

| week      | onderwerp           | boek           | “dictaat”           |
|-----------|---------------------|----------------|---------------------|
| 2–6 sep   | Introductie         | 1              | 1,2                 |
| 9–13 sep  | Types               |                | 3.1/3,3.9,<br>op1/5 |
| 16–20 sep | Controlestructuren  | 2              | 3.4/5,op6/10        |
| 23–27 sep | Funcities,<br>files | 3,4,<br>12.1/2 | 3.6/7,op11/17       |
| 3&4 okt   | geen (werk)college  |                |                     |
| 7–11 okt  | Vervolg, Life       | 6,7.1          | 3.11,op18/25        |
| 14–18 okt | OOP, arrays         | 5              | 3.8,op26/30         |
| ...       |                     |                |                     |

op = “papieren” opgaven van website (“Handouts”); zelf maken, antwoorden: zie website.

In rood: de weken met een deadline op de maandag erna.

- donderdag 16 januari 2025, 13:00–16:00 uur  
Gorlaeus BW.0.17/18/29/30/31/40
- hertentamen: donderdag 27 maart 2025, 13:00–16:00 uur

Tijdig aanmelden via MyStudyMap!

[illegible]

We maken gebruik van het volgende boek:

W. Savitch  
Absolute C++  
sixth edition  
Addison-Wesley, 2016

Oudere drukken zijn ook goed (ouderejaars!). En er zijn vele andere boeken.



En het “dictaat” ([hier](#)) en de sheets ([daar](#)).

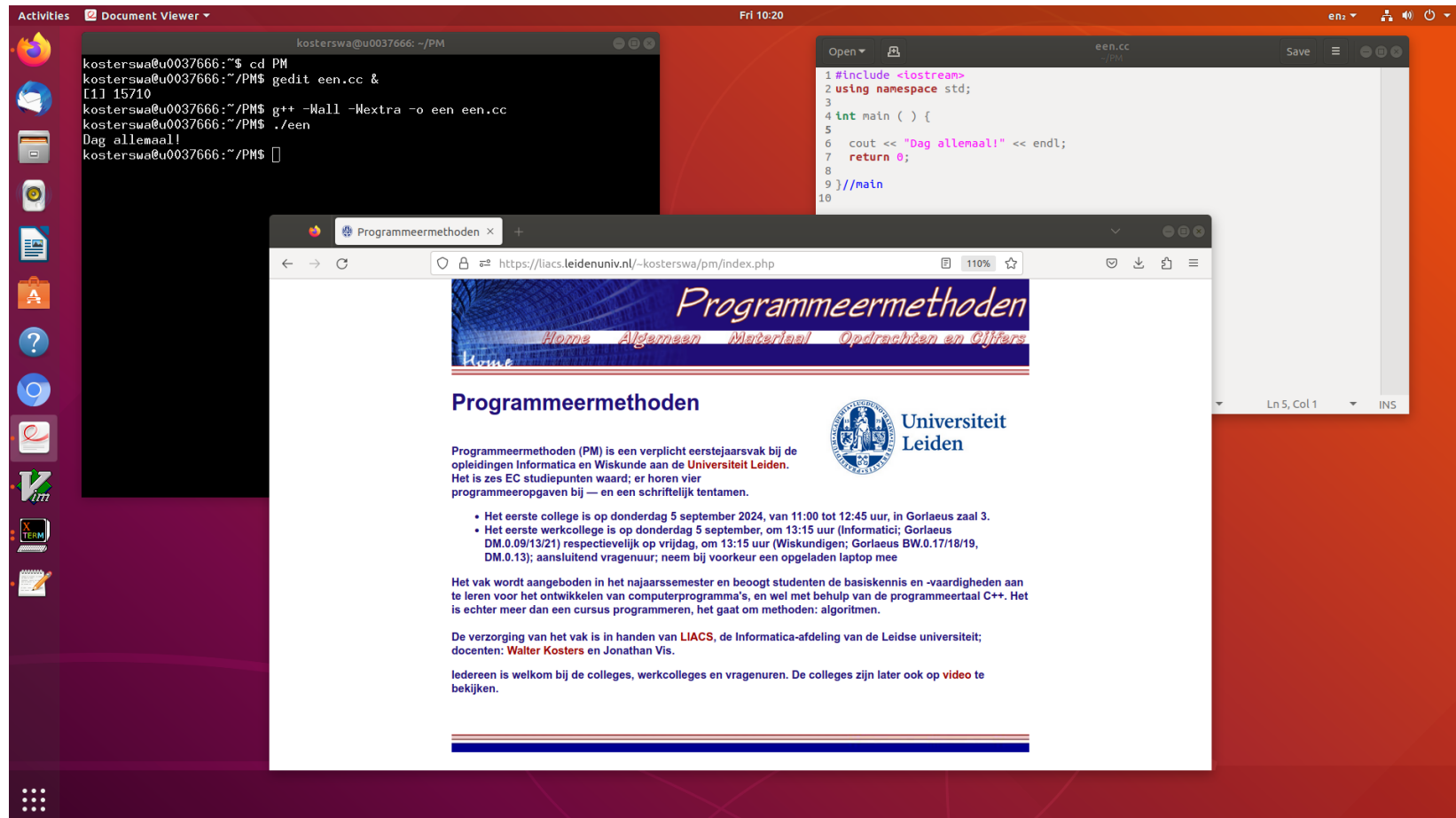
Naast **Microsoft Windows** kunnen PCs ook draaien onder **Linux**, een **operating systeem** (OS) uit de UNIX-wereld (vergelijk Android). In Windows 10 en 11 heb je **Windows Subsystem for Linux** (WSL) — en dat gebruiken we.

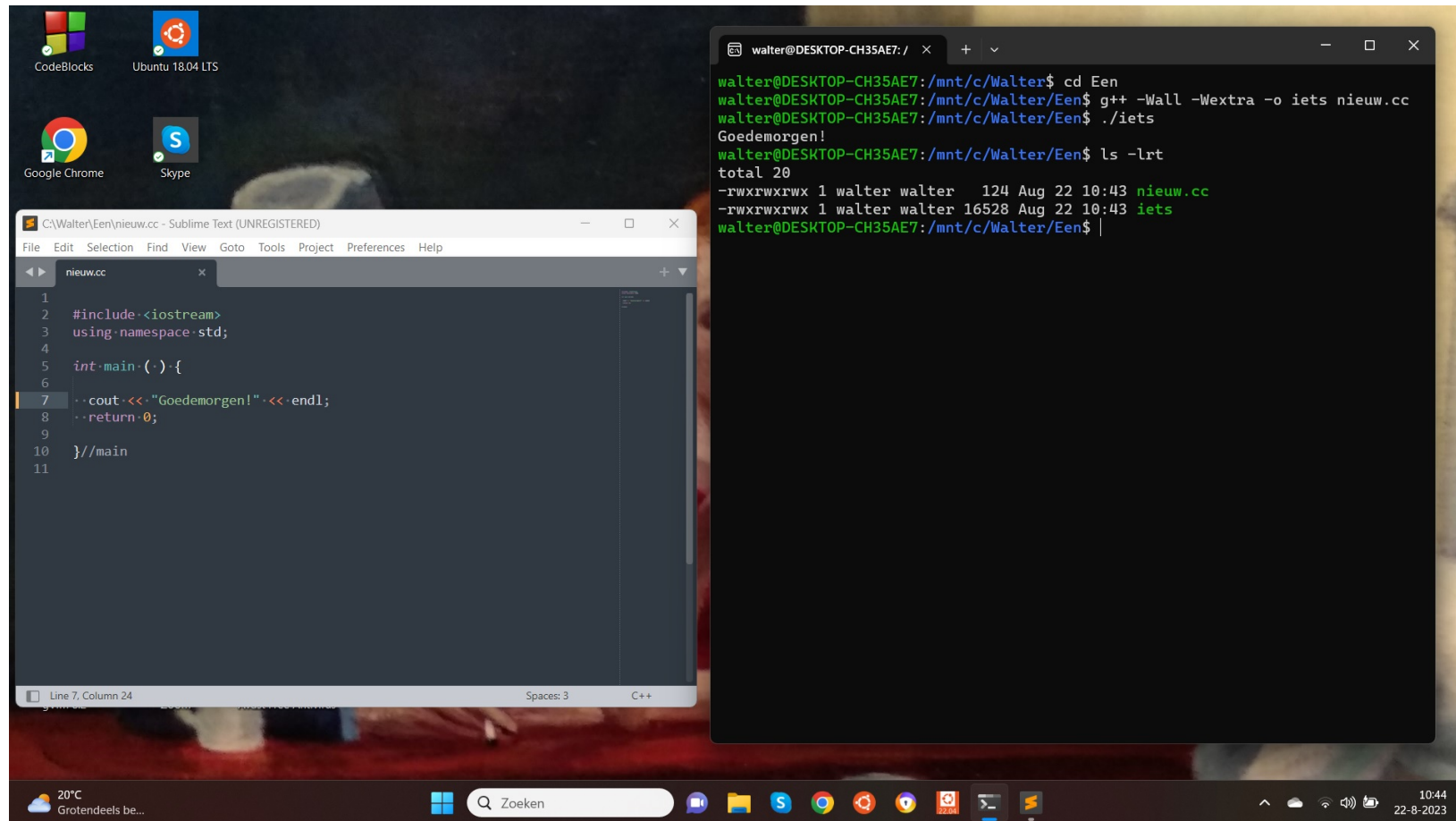
Met een Ubuntu Live-CD/DVD/USB start je PC meteen op in GNOME, een grafische windows-omgeving bovenop Linux. Beter: dual boot. Let er op dat je op een verstandige plaats (USB/e-mail) moet saven. Je kunt ook “gratis” distributies als SUSE, Fedora, Debian, . . . gebruiken — maar dan moet je er meer vanaf weten.



Als je thuis gratis C++ wilt doen, zijn de mogelijkheden:

- ⇒
- Windows: installeer Windows Subsystem for Linux (WSL2, zie straks) en eventueel editor [Sublime Text](#)
  - Mac: zie [hier](#)  
[Mac-video](#)
  - Linux: haal **Ubuntu** van [www.ubuntu.org](http://www.ubuntu.org)  
huidige versie: 24.04 (in computerzalen: 22.04)  
gebruik editor **gedit** en compiler **g++**  
[Linux-video](#)
  - (Windows: **Code::Blocks**, zie straks)





- Installeer **Windows Subsystem for Linux (WSL)** in Windows 11 (of 10), via de “Microsoft store” of `cmd`, `wsl --install -d Ubuntu-24.04` (zie bijvoorbeeld [YouTube](#)).  
in “features”: Virtual Windows Platform en WSL; en in BIOS: virtualisatie
- Open een **terminal** en installeer **compiler** `g++` met `sudo apt install g++` (doe altijd eerst `sudo apt update`).
- Gebruik in Linux `gedit` of een Windows-**editor** zoals Sublime Text.
- Waar zijn je **mappen** = **directories**? Geef het commando `cd /mnt/c/naamvanmap` en zoeken maar ...
- **Hello world** ..., zie straks.

Je kunt ook **Code::Blocks** in Windows gebruiken, maar liever gedit (of Sublime Text) en g++ in Linux (**WSL**).

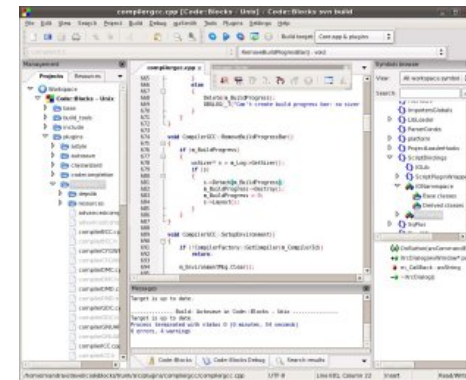
[video](#)

- installeer Code::Blocks van [www.codeblocks.org](http://www.codeblocks.org)  
file: codeblocks-20.03mingw-setup.exe

- zet “warnings” aan!

- edit, compileer en run  
helloworld.cc (let op .cc)

- behandel een eventuele lastige firewall



Een werkend C++-programma maken gaat als volgt:

1. Tik in een **editor** C++-code, het “bron-programma”.
2. Compileer (en link) dit met een **compiler**. Deze vertaalt C++-code naar machinetaal.  
Eventuele fouten: “compile-time-fouten” / “syntax-errors”: “inklude” in plaats van “include”.
3. Draai (= run) deze **executable** vanuit het OS.  
Eventuele fouten: “run-time-fouten”: deel door 0.

Herhaal deze cyclus zo vaak als nodig.

In Code::Blocks vinden alle stappen plaats in een Graphical User Interface (GUI).

(Er zijn ook nog logische fouten.)

Een eerste C++-programma, helloworld.cc geheten:

```
#include <iostream>
using namespace std;
int main ( ) {
    cout << "Dag allemaal!" << endl;
    return 0;
} //main
```

Dit programma zet alleen een tekstje op het beeldscherm.

Let op de — vooral voor mensen nuttige — layout.  
En op hoofdletters en kleine letters.

Een tweede C++-programma (met **syntax-highlighting**):

```
1 // dit is een simpel programma
2 #include <iostream>
3 using namespace std;
4
5 int main ( ) {
6     int getal = 42;    // een variabele declareren
7                       // en initialiseren
8     cout << "Geef een geheel getal .. ";
9     cin >> getal;
10    cout << "Kwadraat is: "
11         << getal * getal << endl;
12    return 0;
13 } //main
```

```
// Dit is een regel met commentaar ...
#include <iostream> // moet er altijd bij
using namespace std;
const double PIE = 3.14159; // een constante (of cmath)
int main ( ) {
    int straal; // straal van de cirkel, geheel getal
    cout << "Geef straal, daarna Enter .. ";
    cin >> straal;
    if ( straal > 0 )
        cout << "Oppervlakte "
                << PIE * straal * straal << endl;

    else
        cout << "Niet zo negatief ..." << endl;
        cout << "Einde van dit programma." << endl;
    return 0;
} //main
```

```
// Dit is een regel met commentaar ...
#include <iostream> // moet er altijd bij
using namespace std;
const double PIE = 3.14159; // een constante (of cmath)
int main ( ) {
    int straal; // straal van de cirkel, geheel getal
    cout << "Geef straal, daarna Enter .. ";
    cin >> straal;
    if ( straal > 0 ) {          // accolades nodig!
        cout << "Oppervlakte ";
        cout << PIE * straal * straal << endl;
    }//if
    else // hier accolades niet nodig, maar liever wel!
        cout << "Niet zo negatief ..." << endl;
    cout << "Einde van dit programma." << endl;
    return 0;
}//main
```



## Programmeermethoden 2024 Eerste programmeeropgave: Breuken

De eerste programmeeropgave van het vak **Programmeermethoden** in het najaar van 2024 heet *Breuken*; zie ook het **eerste**, **tweede** en **derde** werkcollege.

Deze opgave probeert te bepalen of iemand geschikt is voor een studie aan de universiteit. Daartoe moeten enkele vragen beantwoord worden, zo moet je als kandidaat weten op welke dag je geboren bent. En als je niets van breuken weet, is een beta-studie misschien niet verstandig.

Om te beginnen moet de gebruiker diens geboortjaar als geheel getal invoeren, en daarna de geboortemaand, ook als geheel getal. Vervolgens voert men de geboortedag in, wederom als geheel getal. Het programma berekent dan de leeftijd, zowel in aantal jaren als in maanden (bijvoorbeeld: 10 jaar en 3 maanden; 123 maanden); beide worden op het beeldscherm getoond. De leeftijd in maanden wordt analoog aan die in jaren bepaald (als je op de 31ste geboren bent, wordt je iedere maand een maand ouder, maar je bent niet zo vaak "maandig" — dat ben je namelijk alleen op iedere 31ste). Jarige en maandige gebruikers worden gefeliciteerd. Aangenomen mag worden dat het programma op de peildatum 23 september 2024 draait (gebruik `const`; **liefhebbers** mogen met `ctime` de echte huidige dag opvragen en gebruiken). Let op: het programma moet in principe ook op andere peildata vanaf heden tot 2100 correct werken! Gebruikers jonger dan 10 jaar (de 10-de verjaardag nog niet gevierd) of ouder dan 100 jaar (dus 101-ste verjaardag reeds gevierd) worden meteen geweigerd. Als uit het geboortjaar direct al duidelijk is dat het met de leeftijd niets gaat worden, hoeven de vragen naar maand en/of dag niet gesteld te worden. Maar soms biedt pas de dag uitsluitsel!

Nu moet de gebruiker diens geboortedag (zondag, maandag, ..., zaterdag) weten. Als deze fout is, wordt men meteen "verwijderd", en stopt het programma. Het antwoord moet met *één letter* (de eerste letter van de dag; geen cijfer dus) worden gegeven, bijvoorbeeld `w` voor woensdag. In het geval van `d/z` wordt nog om de tweede letter gevraagd.

Het is *niet* de bedoeling `ctime` te gebruiken om deze dag uit te rekenen. Het programma moet een zelf bedachte berekening bevatten om deze dag te bepalen! Gebruik bijvoorbeeld dat 1 januari 1901 op een dinsdag viel. Gebruik *niet* het **Doomsday algoritme** (zie ook **hier**), en ook *niet* allerlei ingewikkelde formules. Voor de periode 1901–2099 geldt dat een jaar een schrikkeljaar is precies dan als het jaartal door 4 deelbaar is.

De echte test bestaat uit enkele vragen. Mensen van 30 jaar of ouder worden hierbij minstens twee maal "netter" aangesproken dan jongeren. Splits de C++-code in het programma niet onnodig vaak!

Er wordt gekeken of de aanstaande student breuken bij elkaar kan optellen of van elkaar kan aftrekken. Wiskundig inzicht is namelijk vereist voor een beta-studie. Mocht dat niet zo zijn, wordt er getest hoe het met de kunst- of literatuurkennis staat.

De gebruiker moet twee willekeurige gegeven breuken, met tellers en noemers tussen 1 en 19, bij elkaar optellen of van elkaar aftrekken (dat laatste staat erbij vermeld). De gebruiker moet het antwoord twee maal geven: als twee gehele getallen (het eerste getal kan negatief zijn; de breuk die hierdoor wordt voorgesteld moet in principe goed zijn; er mag hierbij niet vereenvoudigd worden, uitgaande van het "gewone" optel-algoritme) en als decimaal getal, met een decimale punt, dat er maximaal een constante MAXFOUT (zeg 0.1) naast mag zitten. Als de originele getallen bijvoorbeeld  $1/7$  en  $2/3$  zijn, is het eerste antwoord 17 en 21 (voorstellende de breuk  $17/21$ ), en wordt bij het tweede 0.81 goed gerekend. Nog een voorbeeld:  $2/2$  min  $6/3$  wordt  $-6/6$ . Converteer zelf op de juiste manier tussen int en double. Liefhebbers mogen de resulterende

$$\frac{1}{7} + \frac{2}{3} = \frac{17}{21} \approx 0.81$$

breuk vereenvoudigen (zie het vierde college).

Voor het fabriceren van willekeurige gehele getallen moet gebruik worden gemaakt van de random-generator uit C++. Gebruik bijvoorbeeld `x = rand() % 20`; om een "willekeurig" getal tussen 0 en 19 (grenzen inbegrepen) in de int variabele `x` te krijgen. Zet bovenaan in `main`: `srand(42)`; of `srand(jaar)`; (nadat jaar een waarde heeft gekregen), om de random-generator eenmalig te initialiseren. In plaats van 42 mag ook een ander getal staan — of zelfs, voor liefhebbers, de tijd. En soms is hiervoor `#include <cstdlib>` nodig, helemaal bovenaan het programma.

Is het antwoord goed, dan wordt de kandidaat tot een exacte studie toegelaten, en stopt het programma. Anders wordt één meerkeuzevraag (Aa/Bb/Cc/Dd) over kunst of literatuur gesteld, die uitsluitend biedt over de toelating tot een alpha-studie. Als het daar ook mis gaat, is men helaas niet geschikt voor een universitaire studie. Gebruikers tot of tot en met (kies zelf) 30 jaar krijgen hier een andere vraag dan de oudere gebruikers — maar bij beiden is "hetzelfde" antwoord, bijvoorbeeld steeds B, goed. Wederom, of het antwoord goed of fout is, het juiste antwoord wordt steeds op het scherm afgedrukt.

### Opmerkingen

Als de gebruiker een niet bestaande maand invoert, bijvoorbeeld `-8`, of een jaartal als `4242` (in de toekomst dus), stopt het programma met de mededeling dat dit niet kan (gebruik `return 1;`). Evenzo voor een niet bestaande dag, bijvoorbeeld 31 april of 42 december. We nemen aan dat de gebruiker zo vriendelijk is verder geen fouten te maken bij het invoeren van gegevens: men voert niet al te gekke getallen of letters in, etcetera. Vanzelfsprekend worden wel duidelijke vragen gepresenteerd.

Elk programma moet bij het "runnen" aan het begin op het beeldscherm laten zien wie de makers zijn, wat hun jaar van aankomst, studierichting en studentnummer is, welke opgave het is, wat de gebruiker te wachten en te doen staat, de datum waarop het programma gemaakt is, enzovoorts. Dit noemen we het **infoblokje**. Probeer dit er netjes uit te laten zien. Maak geen al te complexe kaders eromheen; gebruik alleen de eerste 128 gewone karakters.

Bovenaan het programma (in de C++-code dus) staat uiteraard commentaar, waarin een aantal van deze elementen ook weer terugkomen, maar dan meer gericht op programmeurs, bijvoorbeeld de naam van de gebruikte compiler.

Denk aan het gebruik van lege regels, inspringen, commentaar, constanten, enzovoorts. Bovenaan het programma dient zoals gezegd commentaar over het programma te staan, speciaal bestemd voor andere **programmeurs** (en nakijkers), bijvoorbeeld kort wat het programma doet, en welke compiler gebruikt is: gebruikers van het programma vinden dat laatste niet interessant. Het infoblokje moet tijdens het "runnen" van het programma op het scherm komen, en is bestemd voor **gebruikers** van het programma. Lees ook eens over **richtlijnen** bij het maken van programmeeropgaven, en bestudeer de **huisregels**. Er hoeft geen gebruik van functies, arrays en het `while`- en `for`-statement gemaakt te worden. Alleen de headerfiles `iostream` mag en moet gebruikt worden — en eventueel `ctime` voor liefhebbers; en misschien `cstdlib` voor het gebruik van de random-generator. Ruwe indicatie voor de lengte van het C++-programma: 200 regels (300 mag ook wel). Letters moeten als `char` worden ingelezen, dus *niet* met strings, die mogen namelijk niet gebruikt worden.

Uiterste inleverdatum: **maandag 23 september 2024, 18:00 uur**.

De manier van inleveren (één exemplaar per koppel, dat — ter herinnering — uit maximaal twee personen bestaat) is als volgt.

- Digitaal de C++-code inleveren via Brightspace > Course Tools > Assignments. Stuur geen executable's, LaTeX-files of PDF-files, lever alleen één C++-file digitaal in!
- Doe een print van het verslag in de doos bij kamer Gorlaeus BM.2.07.

De laatste voor de deadline ingeleverde versie wordt nagekeken.

Overal duidelijk datum en namen van de twee makers vermelden, in het bijzonder als commentaar in de eerste regels van de C++-code. Lees bij het **derde werkcollege** hoe het verslag eruit moet zien. Zijn spaties/tabs goed verwerkt?

Te gebruiken compiler: als hij maar C++ vertaalt; het programma moet in principe zowel op een Linux-machine (met g++) als onder Windows met Code::Blocks draaien. Test dus zo mogelijk op beide systemen! Normering: (consequente) layout 2; commentaar 2; infoblokje 1; verslag 1; werking 4.

Eventuele **aanvullingen en verbeteringen**: lees de WWW-bladzijde die je nu ziet:

[www.liacs.leidenuniv.nl/~kosterswa/pm/op1pm.php](http://www.liacs.leidenuniv.nl/~kosterswa/pm/op1pm.php). Recente aanvullingen en/of wijzigingen staan in **rood**.

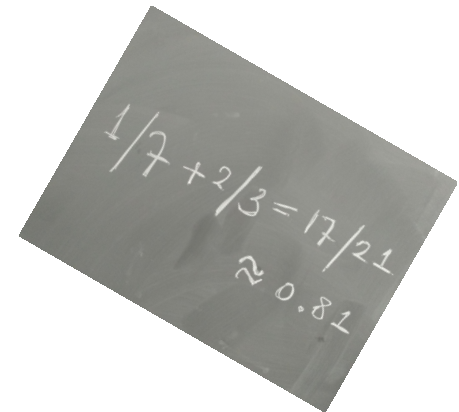
Vragen: e-mail naar [pm@liacs.leidenuniv.nl](mailto:pm@liacs.leidenuniv.nl)

[www.liacs.leidenuniv.nl/~kosterswa/pm/op1pm.php](http://www.liacs.leidenuniv.nl/~kosterswa/pm/op1pm.php)

Voor de eerste programmeeropgave moet je onder meer, voor een gegeven datum, de dag van de week uitrekenen. Bijvoorbeeld: 1-1-1901  $\rightarrow$  dinsdag.

Hoe zou je dat uit je hoofd doen?

Wanneer is een jaar een schrikkeljaar?



[www.liacs.leidenuniv.nl/~kosterswa/pm/op1pm.php](http://www.liacs.leidenuniv.nl/~kosterswa/pm/op1pm.php)

Verslag in **L<sup>A</sup>T<sub>E</sub>X**! En wat is  $1/7 + 2/3$ ?

Niet gebruiken: Doomsday-algoritme

5-9: donderdag

4-4, 6-6, 8-8, 10-10, 12-12, "I work from 9-5 in a 7-11".

In een [Linux](#) windows-omgeving zoals WSL start je een of meer **terminals**: windows waarin je tekst-georiënteerde opdrachten kunt geven. Daarin tik je in:

```
gedit een.cc &
```

```
g++ -Wall -o een een.cc  
./een
```

```
ls -lrt
```

edit je eerste C++-programma;  
open hiertoe een “edit-window”  
compileer een.cc naar een  
run de executable een  
overzicht van je files (%)

De ampersand & zorgt er voor dat je in het oorspronkelijke window ook kunt doorwerken. En (%) levert zoiets als:

```
-rw-r--r-- 1 kosterswa domain users 124 Sep 5 12:42 een.cc  
-rwx----- 1 kosterswa domain users 11049 Sep 5 12:43 een
```

- donderdag 13:15–... (Informatici; DM.0.09/13/21)
- met aansluitend vragenuur
- vrijdag 13:15–... (Wiskundigen; BW.0.17/18/19; DM.0.13)
- met aansluitend vragenuur
- (bij voorkeur) op eigen laptop met WSL
- [www.liacs.leidenuniv.nl/~kosterswa/pm/pmwc1.php](http://www.liacs.leidenuniv.nl/~kosterswa/pm/pmwc1.php)
- doel: compiler, Hello world, int, opgave, ...

- Hello world voor C++
- boek en “dictaat”
- ULCN-account
- software voor thuis/laptop: WSL, ...
- de eerste programmeeropgave
- [www.liacs.leidenuniv.nl/~kosterswa/pm/](http://www.liacs.leidenuniv.nl/~kosterswa/pm/)
- huiswerk: Savitch Hoofdstuk 1; dictaat 1 en 2