Richtlijnen, tips en beoordelingscriteria voor de huistaken en het project van TWS

Peter Opsomer *22 september 2015*

Inleiding

Deze criteria zijn gebaseerd op het feedbackrooster [1], de tips in [2], de richtlijnen van vorig jaar, de 'factoren beoordeling project' in [3] en algemene opmerkingen bij inzendingen van vorige jaren. Deze kunnen natuurlijk opgeheven worden door andere mededelingen. Bij de beoordelingscriteria verder in dit document wordt hetgeen optioneel is, aangeduid met vierkante haakjes.

Opgaves voor de projecten worden beschikbaar gemaakt tijdens de tweede, derde of vierde oefenzitting voor deel 1 en voor deel 2 kort na het indienen van het eerste deel (ongeveer in de helft van het semester) met deadline ongeveer anderhalve week vóór de mondelinge besprekingen, die op voorhand gepland worden op Toledo.

Opstellen en activeren Fortran compilers

Log in met Ubuntu, zoek (icoon linksboven) naar 'terminal', sleep die naar de taakbalk en klik.

Open `~/.options.sh` (via `$ gedit ~/.options.sh` in de terminal of bij mapweergave Edit-Preferences-Views-Show hidden and backup files) en voeg `CS_FORTRAN=yes` toe. Herstart eventueel de terminal.

of

Druk `Ctrl-Alt-F1` en log in op de tekst console. Voer het commando `echo CS_FORTRAN=yes >> ~/.options.sh` uit en log uit (`logout`). Druk `Ctrl-Alt-F7` of `Ctrl-Alt-F8` en log in op het grafisch scherm.

ssh

Je kan een eigen laptop gebruiken; de nodige compilers hierop installeren (ifort is gratis voor studenten via `https://software.intel.com/en-us/qualify-for-free-software/student`) zal echter moeilijk zijn. Dus werk dan liever met ssh. Je kan in de SoftwareOntwerpLabo's op een eigen laptop of van thuis uit in Linux werken met bijvoorbeeld `ssh s0000000@ssh.cs.kuleuven.be` of met `st.cs.kuleuven.be` en dan `ssh -X -l s0000000 Y.cs.kotnet.kuleuven.be` met Y een naam van een machine in de computerklassen. In Windows kan je bijvoorbeeld Putty installeren. Als paswoord neem je hetzij hetgeen hoort bij je studentennummer `s0000000`, hetzij één die je aanmaakt: zie de documentatie van systeembeheer, vb `http://system.cs.kuleuven.be/cs/system/wegwijs/computerklas/internet/`.

Voor grafische toepassingen (vb gedit) moet je X11 forwarding enabelen en vb XMing telkens opstarten. Dit is echter traag; je kan `vim bestand.f95` gebruiken om in de terminal te werken. Als je denkt om met ssh te gaan werken, probeer dit dan op tijd (niet één dag voor een deadline) werkende te krijgen met behulp van de documentatie van systeembeheer, iemand bij wie het werkt, ...

En, met dank aan Michiel: *Om een ssh-verbinding tot stand te brengen met de Pc-klas, is het handig om deze tutorial te volgen: `http://dietercastel.com/2013/03/02/ssh-to-computer-science/`. Hier staat onder andere duidelijk hoe je een keypair genereert en dan eerst met een bufferserver moet verbinden om van daaruit met een computer uit de Pc-klas te verbinden.*

De scripts voor het doorsturen van bestanden van en naar uw eigen map op de server heb ik niet getest. In plaats van deze scripts, gebruik ik het programma MobaXterm ter vervanging van Putty om de ssh-verbinding te maken. Je kan hierin een nieuwe sessie starten door op session→ssh te klikken en dan hetzelfde te doen als de blog aangeeft voor Putty (eerst verbinden met de bufferserver st.cs.kuleuven.be, dan met een Pc-klas computer via het commando: ssh $rxxxxxx@orval.cs.kotnet.kuleuven.be of ter vervanging van orval een andere Pc-klas computer.) Je kan ook gewoon een opgeslagen Putty-profiel laden om te verbinden met de bufferserver zodat MobaXterm via Putty meteen uw private key vindt.

Om bestanden door te sturen, start je ook een nieuwe sessie, maar kies je SFTP. Dan vul je uw r-nummer in als gebruikersnaam en als host neem je bvb: orval.cs.kotnet.kuleuven.be. Een andere naam van een Pc-klas computer werkt uiteraard ook. Je kan dan gewoon bestanden verslepen tussen uw eigen computer, links in de balk, en de server rechts.

Nuttige terminalcommando's

`$ cd ~/`, `$ cd ..`, `$ ls`, `pwd`, `$ man {commando}` , `$ {commando} --version`, `$ pdflatex source.tex`, `$ make`, `$ make r FC=g95`, `$ mv file dir/`, `$ cp file dir/`, `$ mkdir dir`, `$ top`, `$ locate hello`, ↑, Tab (automatische aanvulling), Alt-Tab, Ctrl-s, Ctrl-c (onderbreking van programma-uitvoering), Ctrl(-Shift)-c/v, Alt-↑, `$ time {prog}`, `$ valgrind {prog}`, `$ cat {in} | {prog}`, `$ ./prog < {in} > {out}`

Indienen huistaken

Voor de oefenzittingen Fortran, Matlab en Python dien je de elektronische versie(s) van je bronbestand(en) in die gevraagd worden in de opgave voor de huistaak: zie verder. De gevraagde figu(u)r(en) voeg je eveneens toe als bijlage, bij voorkeur in rasterformaat. Zorg voor andere bestandsnamen dan vorige inzendingen en stuur geen `*.zip` of equivalent. Mail deze met onderwerp ("TWS Oefenzitting x") naar `peter.opsomer@cs.kuleuven.be`. Je hoeft dus geen verslag in pdf te maken (tenzij expliciet gevraagd in de opgave voor die huistaak). De manier van indienen voor de oefenzittingen C++ zal bij de opgave staan.

De deadline voor het indienen van de elektronische versie van de huistaken is meestal 14u00 op *de dag voorafgaand aan de volgende TWS-oefenzitting*. In bepaalde gevallen kan dit dus na twee weken zijn of meer. Deadlines zijn strikt, zowel nu als in je latere beroepsleven; zie bijvoorbeeld ook 'Taken en project-Varia-Deadline'.

Voeg voor elk bestand het volgende toe in commentaar bovenaan:

- Je naam.
- Met welke compiler(s) (+ versie-nummer) je je programma getest hebt.
- Welk compiler-commando (bij voorkeur één) en welke compiler-opties er gebruikt moeten worden om je programma te compileren. Een gestructureerde `Makefile` kan ook als extra bijlage wanneer het veel commando's betreft (> 3 per compiler).
- Antwoorden en uitvoer voor de vragen, in commentaar bovenaan het respectievelijke bronbestand.
- Zie 'Taken en project'.

Bijkomend voor het eerste bestand in bijlage:

- Hoeveel tijd je in totaal aan de huistaak zelf hebt besteed, eventueel apart ook aan het voorbereiden van de oefenzitting, maar niet aan (het inhalen van) de oefenzitting zelf. Probeer dus tijdens het maken van de taak een schatting bij te houden.
- Bij voorkeur is dit het hoofdprogramma.

Taken en project

Aanvulling regels

Alle huistaken en beide delen van het project voor het OPO Technische-Wetenschappelijke Software worden gequoteerd, en het examenreglement is dan ook van toepassing. Volledigheidshalve beschrijven we hieronder wat hierbij wel of niet toegestaan is op het vlak van samenwerking.

De oplossing en/of verslag en/of programmacode die ingediend worden, moeten volledig het resultaat zijn van werk dat je zelf gepresteerd hebt. Je mag je werk uiteraard bespreken met andere studenten, in de zin dat je praat over algemene oplossingsmethoden of algoritmen, maar de bespreking mag niet gaan over specifieke code of verslagtekst die je aan het schrijven bent, noch over specifieke resultaten die je wenst in te dienen. Als je het met anderen over je taken of project hebt, mag dit er dus NOOIT toe leiden, dat je op om het even welk moment in het bezit bent van een geheel of gedeeltelijke kopie van de opgeloste taak, project of verslag van anderen, onafhankelijk van het feit of die code of verslag nu op papier staat of in elektronische vorm beschikbaar is, en onafhankelijk van wie de code of het verslag geschreven heeft (medestudenten, eventueel uit andere studiejaren, volledige buitenstaanders, internet-bronnen, e.d.). Dit houdt tevens ook in dat er geen enkele geldige reden is om je code of verslag door te geven aan medestudenten, noch om dit beschikbaar te stellen via publiek bereikbare directories of websites.

Elke student is verantwoordelijk voor de code en het werk dat hij of zij indient. Als tijdens de beoordeling van een huistaak of project er twijfels zijn over het feit of dit zelf gemaakt is (bv. gelijkaardige code, grafieken, tekst of resultaten), zal de student gevraagd worden hiervoor een verklaring te geven. Indien dit de twijfels niet wegwerkt, zal dit gemeld worden als een onregelmatigheid, zoals voorzien in het onderwijs- en examenreglement (zie `http://www.kuleuven.be/onderwijs/oer/`).

Ronald Cools, Karl Meerbergen & Dirk Nuyens
P.S. Plagiaatdetectiesoftware kan aangewend worden om problemen te detecteren.

Code

IO Bestandsformaat/-structuur zoals gevraagd.
Inlezen en uitschrijven: zoals gespecifieerd en niet te veel.
Print '!' voor resultaten zodat je output gewoon kunt copypasten naar bovenaan je bronbestand voor taken.

Precisie Kies een geschikte soort, het best parametriseren, vermijd compilerafhankelijke definities.

Code Interface van je (sub)routines is vrij te kiezen.
Documenteer kort slechts als het niet triviaal is en de leesbaarheid van je code verbetert.
In elk bestand bovenaan je naam en korte beschrijving.
Dealloceren, overal `implicit none`.
Volledigheid: implementatie van alle specificaties.

Stijl Zie voorbeeldcode die op Toledo komt op het einde van de tweede oefenzitting, 'Scientific Software Development with Fortran', de Fortran standaard zelf, opgaves van de oefenzittingen en gegeven voorbeelden.
Op zich niet zo belangrijk, maar wees consequent.
Overzichtelijk: Logische structuur, geen gecommentarieerde of gedupliceerde code (niet te veel copypasten), vermijd te 'objectgeoriënteerd' (veel types en kleine functies + moeilijk leesbaar en moeilijk om effectieve berekeningen terug te vinden, ...), uitlijning/indentatie, leesbaarheid, ...

Performantie [Snelheid (CPU, wallclock, ...), geheugen, nauwkeurigheid (maximumnorm, elementgewijs, 2-norm, ...), convergentiesnelheid, complexiteit, paralleliseerbaarheid, uitbreidbaarheid, leesbaarheid, installatiegemak, ...]
[Bespreking van caching-effecten, loop unrolling, spatial en temporal locality, prefetching.]
[Cijfermateriaal en/of grafieken met hun bespreking.]
[Optimaliseren van performantie, bijvoorbeeld ook door volgorde van de lussen/bewerkingen te veranderen, hergebruik van geheugen, gebruik van externe bibliotheken zoals BLAS en/of door pass by reference in plaats van pass by value.]
[Gevectoriseerd werken ipv elementsgewijs met forall/indexering/intrinsieke functies/...]
[Statisch ↔ dynamisch geheugen ↔ pointers, stack tegenover heap.]
Korte functies beter inlinen in plaats van aparte routine voor snelheid en soms zelfs uitbreidbaarheid.
Gebruik Lapack/BLAS, zie bijvoorbeeld `http://www.netlib.org/lapack/single/`.
Denk na over het nut en de leesbaarheid van code die je schrijft.

Debuggen Bespreek kort of er problemen waren.
Geef variabelen niet de naam van (intrinsieke) functies/routines.
Fortran maakt (meestal) geen onderscheid tussen hoofdletters en kleine letters.
Floating points getallen niet testen op exacte gelijkheid.
Als je syntax errors krijgt, los dan eerst de eerste op omdat de volgenden daardoor veroorzaakt kunnen zijn.
Initialiseer alle variabelen (bijvoorbeeld op 0) en ga na of het verschil maakt of je dat doet in de definitie.
Een lijn wordt afgekapt op 132 characters.
Vergelijk met een (Matlab-)implementatie of het exacte resultaat.
Test elke verandering vanaf je backup.

Driver Getest op alle compilers: sommige geven een betere performantie of een compiler error in plaats van runtime segmentation faults.

Algoritme

Experimenten Betrouwbare tijdsmetingen + bespreking.
[Reken-/ geheugen-/... complexiteit en eventueel schatting of meting.]
Impact andere factoren bespreken.

Algoritme Uitwerking ontbrekende details, meestal in formulevorm.

Verificatie Test met een simpel voorbeeld.
[Geef genoeg eigen voorbeelden en een niet te korte bespreking: probeer speciale gevallen die proberen je programma te doen falen, ander (kwalitatief) gedrag, ...]
Voer testen (geautomatiseerd als het er veel zijn) uit in de terminal, zodat je fouten in je hoofdprogramma ook kunt opsporen en zodat je code korter is.
[Figuren ter verificatie van het algoritme of evaluatie van performantie: zie 'Visualisatie' in 'Vormgeving en presentatie'.]

Robuustheid Test met een ongekende, onverwachte of speciale input/optie/argument, lege arrays, getallen die nul, negatief en/of zeer afwijkend zijn.
Verhinder dat dit het programma blokkeert, iets doet dat de gebruiker niet verwacht of wacht op input: geef beter een error en stop.
Dit is wel af te wegen tegenover de leesbaarheid en beknoptheid van je code.
[Test ook in je code, vb of bepaalde variabelen `NaN`, groter of kleiner dan verwacht worden.]
Controleer op wiskundige of fysische onmogelijkheden.

Varia

Opmerkingen uit of bij de commentaar bovenaan de code of het verslag.

Makefile (enkel als veel commando's): laat weg wat je niet gebruikt, gebruik geen absolute paden (/Desktop/, /usr/, ...), zorg voor mogelijkheid tot hercompilatie om gevaren tijdens debuggen te vermijden. Wanneer je `call SYSTEM(...)` gebruikt met nagfor, kan je je Makefile 'use f90_unix_proc' laten toevoegen, een kort bestand laten toevoegen of een andere oplossing bedenken.

Vlaggen [Bespreking van compilatievlaggen en hun effect].

Feedback Jullie krijgen ook telkens individuele feedback: probeer daar ook rekening mee te houden bij het maken van volgende huistaken. Deze wordt meestal mondeling gegeven, dus onthoud ze of schrijf eventueel op.
Hou zeker rekening met feedback op het eerste deel van het project bij het maken van het tweede deel; argumenteer toch minstens waarom je iets niet hebt aangepast hieraan. Zorg dan ook ten laatste voor een antwoord op de vragen die in de individuele feedback van het eerste deel van het project aangegeven zijn met een vraagteken, maar je hoeft geen nieuwe versie te maken van het verslag van het eerste deel.

Deadline Alles juist en op tijd ingediend: geen *.o, *.mod,*.out, backups (.f90∼ etc), verborgen bestanden (.gout...), executables, niet-gevraagde bibliotheken, ...
[Om de timing te kunnen respecteren, maak je het best ook backups van werkende versies van je programma, om mee te vergelijken en zodat je daarop kunt terugvallen als je iets verandert waardoor het niet meer naar behoren werkt.]
Taken en projecten die goed op voorhand zijn ingestuurd, bevatten over het algemeen minder fouten. Dit laat ook toe om ieders inzending op tijd verbeterd te krijgen. We nemen ook in beschouwing dat je geen rekening kunt houden met dingen die op het discussieforum verschijnen nadat je hebt ingestuurd.
Spreid je tijdsbesteding.
Probeer niet veel tijd te verliezen met delen die niet zo relevant zijn dus post iets op het discussieforum (bekijk dit regelmatig) als je iets te moeilijk vindt of er te veel tijd aan besteedt.
(Vaak) te laat indienden (zonder gegronde reden) komt ter sprake bij de mondelinge bespreking, zeker bij indienen nadat (algemene) feedback al gegeven is.
Herlees ook telkens de opgave (en deze richtlijnen) vooraleer je instuurt om zeker te zijn dat je niets vergeten bent.

Volledigheid Wanneer je denkt te veel tijd te gaan moeten spenderen aan een deel van de opgave, vraag om verduidelijking op het discussieforum. Als je vast zit op een klein deel, vraag om een oplossing daarvan via mail zodat je ten minste kunt doen wat erop volgt.
Als het je niet lukt om sommige specificaties uit de opgave uit te voeren, mag je die weglaten: spendeer geen uren aan dezelfde kleine fout! Vermeld dat dan wel duidelijk; het is natuurlijk wel nodig om zoveel mogelijk specificaties te implementeren.

Optioneel [Uitvoering van optionele delen, uitbreiding op opgave/oefenzitting.]

Overeenkomst Zorg voor overeenkomst tussen commentaar bovenaan code/het verslag en (uitkomsten van) code.

Twijfel Bij onduidelijkheden in de opgave mag je veronderstellingen nemen die je argumenteert. Als je toch nog twijfelt, kan je natuurlijk vragen stellen, het best op het discussieforum.

Individueel Je mag overleggen met collega's op een conceptueel niveau, maar iedereen maakt de taken en het project individueel. Bij een uitbreiding op de oefenzitting vermeld je ook met wie je hebt samengewerkt voor het deel van de oefenzitting.

Koppel Koppel je persoonlijke aanpak expliciet aan de opgedane kennis in de lessen, oefenzittingen en bij het lezen van het cursusmateriaal.

Enkel voor het project

Een verschil met de huistaken is dat we wel verslag in pdf verwachten voor elk van de twee delen van het project. De structuur voor het indienen daarvan (op Toledo in plaats van via mail), de bronbestanden en eventuele andere bestanden zal bekend gemaakt worden in de opgave van het respectievelijke deel van het project.

Bijkomend voor je code beoordelen we voor **IO** of je `--help` duidelijk is en voor de **Driver** het afhandelen van commandline-argumenten [in willekeurige volgorde]. Je kan ook versiecontrole doen (hoeft niet fancy te zijn) en vermelden in het verslag hoe en waar het nuttig was. Hieronder volgen enkele tips en criteria voor het verslag, waarbij 'Schrijftaal' weinig doorweegt.

Structuur en opbouw

Te vermijden Er is geen titelblad, inhoudsopgave, opsomming van de secties van het verslag, herhaling uit de opgave [of het verslag van het eerste deel] nodig voor dit type verslagen.

Titel De titel is specifiek en geeft de lezer een precies idee van de inhoud.

Abstract Het abstract vormt een samenvatting van de inhoud van het verslag. Dit is 1 alinea over de belangrijkste resultaten, je totale tijdsbesteding, welke extra delen van de opgave je hebt geïmplementeerd, grootste struikelblokken, ...

Methodologie Het middendeel start met een beschrijving van de *uitwerking* van het project waarin ook het belangrijkste werkingsprincipe wordt toegelicht. Link ook expliciet met dit vak TWS, hoorcolleges, oefenzittingen en cursusmateriaal.
Het middendeel gaat dieper in op de gevolgde werkwijze en geeft objectieve argumenten voor gemaakte keuzes. Bespreek dus je softwareontwerp, alternatieve implementaties, uitbreidingen, stukken code waarmee je problemen had, methodiek en aanpak, ook voor het debuggen [en je performantieverbeteringen]. Hou dit dus bij door bijvoorbeeld een kladversie van je verslag te maken.
Vermijd hierbij vage besprekingen zonder specifieke verwijzingen naar je code.
Als een bepaald onderdeel van de opdracht niet gelukt is of als je code niet compileert, geef dit dan duidelijk aan in je verslag en geef aan waar je vermoedt dat het probleem zit.
Tijdens het maken van het project hou je je tijdsbesteding bij en geef ze opgesplitst in 5 à 10 onderdelen in het verslag.

Resultaten en analyses worden systematisch gerapporteerd. Enkel relevante resultaten worden besproken en geïnterpreteerd.
Analyse en (fysische) interpretatie van data zijn volledig en correct. Beperkingen en implicaties voor de interpretatie van de resultaten worden gerapporteerd.
De gevolgde redenering is voldoende diepgaand en genuanceerd en eventuele veronderstellingen worden genoeg gemotiveerd.

Discussie ≈ interpretatie en argumentatie.
Een discussie bespreekt de voornaamste bevindingen. Hoofd- en bijzaken worden goed uit elkaar gehouden.
In de discussie zijn onderbouwing van argumentatie en conclusies consistent en met aandacht voor alternatieve verklaringen.
Als je timings geeft, vermeld dan ook kort de architectuur van de gebruikte machine en of het kan zijn dat een deel van de CPU ingenomen wordt door andere gebruikers.

Besluit Het algemeen besluit vat de voornaamste besluiten of bijdragen samen.
Het besluit eindigt met voorstellen voor verbetering en/of suggesties voor verder onderzoek.

Bijlage (Waarschijnlijk niet nodig) Lange bewijsvoeringen, berekeningen, voorbeeldcode (meer dan ongeveer 10 lijnen achtereen) en uitgebreide experimentele resultaten, die niet essentieel zijn voor de tekst, zijn opgenomen in bijlage.
Elke bijlage draagt een nummer en in de tekst wordt ook naar elke bijlage verwezen.

Schrijftaal

Taal De taal in je codedocumentatie/wat je afprint mag je zelf kiezen maar moet correcte schrijftaal zijn.

Schrijfstijl Een wetenschappelijke tekst is onpersoonlijk, neutraal en hoort ook objectief, formeel en zakelijk geschreven te zijn. Gebruik dus niet te veel de ik-vorm.
De schrijftaal is bondig, met specifieke en kernachtige bewoordingen.
De schrijftaal is duidelijk, leidt tot een goed begrip van de betekenis en ondersteunt de boodschap.

Taalgebruik De tekst bevat correcte en geen onnodig lange zinsconstructies.
Een lange zin is opgedeeld in twee om de leesbaarheid te verhogen.
Een zakelijke tekst is geschreven met een correcte grammatica en spelling. Vermijd dus typo's, bijvoorbeeld door `aspell check -t --lang=nl verslag.tex` of via `ispell`.
De tekst maakt voornamelijk gebruik van actieve werkwoordsvormen in de tegenwoordige tijd met een minimum aan hulpwerkwoorden. Dit is niet altijd mogelijk, maar probeer het.
In de tekst worden eenvoudige hoofd- en rangtelwoorden (tot twaalf) voluit geschreven, behalve als het exacte informatie betreft.
Splitsingstekens (*hyphenation*) moeten OK zijn, zie [2].

Vormgeving en presentatie

Structuur De tekst heeft een logische indeling en volgorde. De onderdelen van de tekst sluiten goed op elkaar aan en zijn gelijkmatig verdeeld.
Het verslag bestaat uit een doorlopende tekst met een overzichtelijke en uniforme lay-out zonder te veel opsommingen.
De tekst bestaat uit paragrafen van ongeveer gelijke lengte. Binnen de paragrafen is er voldoende samenhang, met heldere overgangen tussen alineas aan de hand van witregels zonder indentering.
Een alinea is een samenhangend geheel en het verband tussen verschillende zinnen is duidelijk dankzij signaal-, verbindings-, en verwijswoorden.

Visualisatie Sprekende resultaten worden voorgesteld in tabellen, figuren of grafieken en verduidelijken de tekst. Deze zijn nodig voor het beter beoordelen van de performantie en discussie.
Tabellen, figuren en grafieken dragen een nummer, titel en verklarend onderschrift. Indien nodig is er een bronvermelding voorzien of wordt een legende gegeven.
De begeleidende tekst verwijst steeds en rechtstreeks naar (het nummer van) de tabel, figuur of grafiek. Er wordt ook een interpretatie gegeven.
In een grafiek worden assen benoemd, grootheden en eenheden weergegeven.
Grafieken moeten duidelijk zijn en door verschillende kleuren te mengen met streepjeslijnen enzovoort kan je problemen vermijden bij zwart-wit printen.
Grafieken hebben relevante (lineair, (semi)logaritmisch) en benoemde assen.
Denk bij veel figuren na hoe je ze kunt samenzetten om zo bijvoorbeeld verschillende methodes gemakkellijker te kunnen vergelijken.

LaTeX Gebruik [*usepackage* {] *epstopdf* [}]: extensie niet specifiëren, gebruik vectorformaat.
\usepackage[dutch]{babel} voor Figuur ipv figure + controleer de splitsingspatronen (*hyphenation/textconfig*).
\usepackage{graphicx, subfigure, amsmath, amsfonts, geometry, hyperref}.
Gebruik eventueel (re)newcommand, align[*], eqref, bibtex/thebibliography, \everymath{\displaystyle}.

Formules Eenheden van grootheden en de tijdsmetingen worden toegevoegd.
Een symbool wordt steeds in cursief geschreven.
Er is een consequent gebruik van dezelfde standaardsymbolen en -notaties.
Formules of vergelijkingen worden met een vergelijkingseditor in de tekst geplaatst.
Een lange formule of vergelijking staat op een aparte regel in de tekst.
Het aantal getoonde beduidende cijfers is enkel hoog als het nodig/relevant is.
Toon enkel een formulenummer als je ernaar verwijst in de tekst; zet het daar tussen haakjes of doe \eqref.

Referenties In een verslag zoals hier is het niet echt nodig om een aparte bibliografielijst aan te maken, afhankelijk van het aantal referenties.
Geef referenties voor wat je hebt gebruikt en niet 'algemeen gekend' is. Refereer zeker naar originele code als je het aangepast/gebruikt hebt (voor een niet-essentieel deel van het project !).
Elke referentie bevat alle nodige informatie om ze op te zoeken.

Omvang De tekst voldoet aan de vooropgestelde omvangvereisten.
De tekst is bondig maar volledig zonder al te brede marges.

Referenties

[1] Faculteit Ingenieurswetenschappen. Feedback rooster schriftelijk rapporteren. `https://eng.kuleuven.be/onderwijs/rapporteren/feedbackrooster-schriftelijk-rapporteren-2.pdf`, april 2014.

[2] D. Nuyens. Thesis layout in LaTeX. `http://people.cs.kuleuven.be/~dirk.nuyens/thesislayout/`, april 2014.

[3] K. Poppe. KUL Toledo TWS/Project/Factoren beoordeling project, H03F0a: Project. output.html.