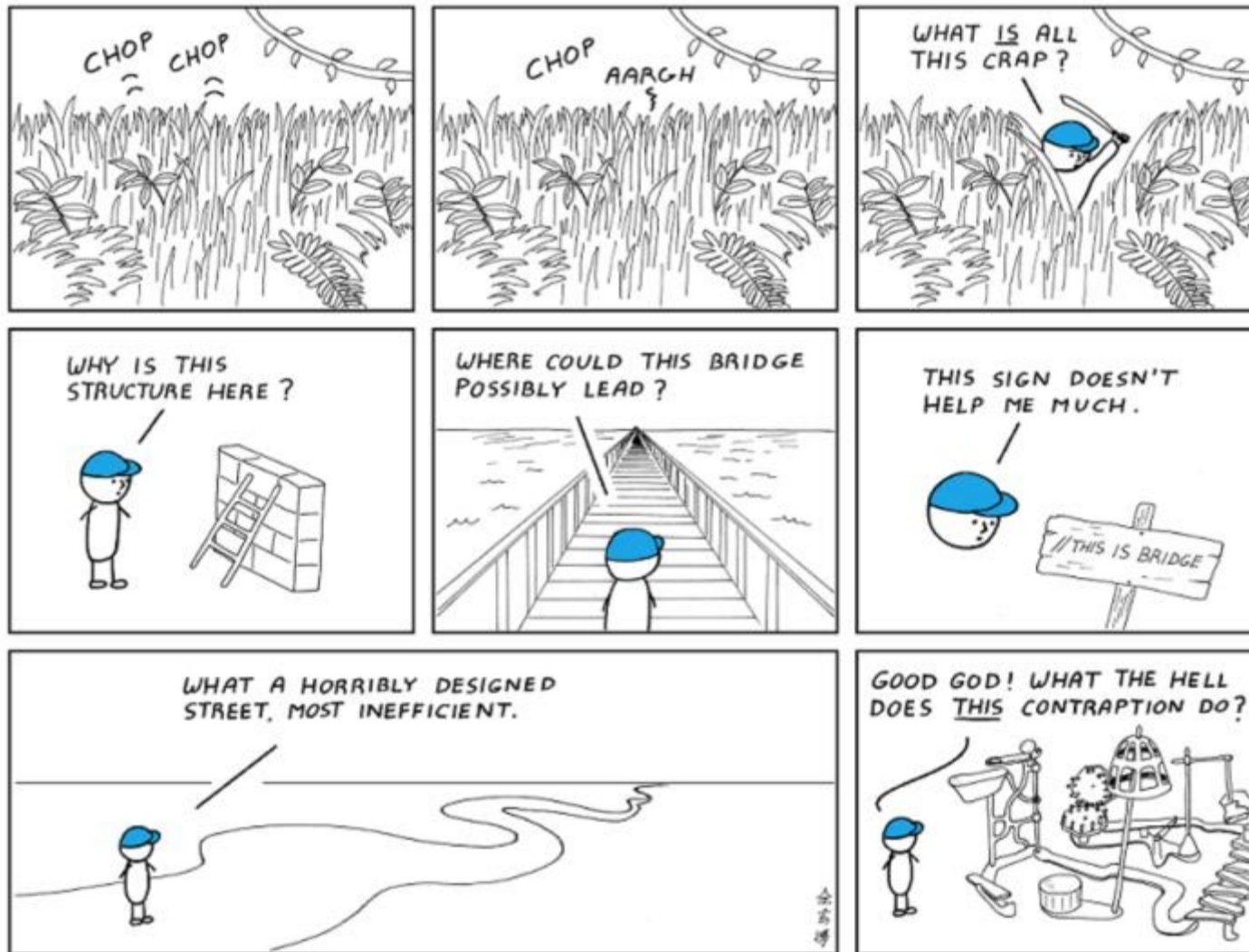


**EMS31 Kwartaal 3 Week 6:
Coding standards en static code analysis**

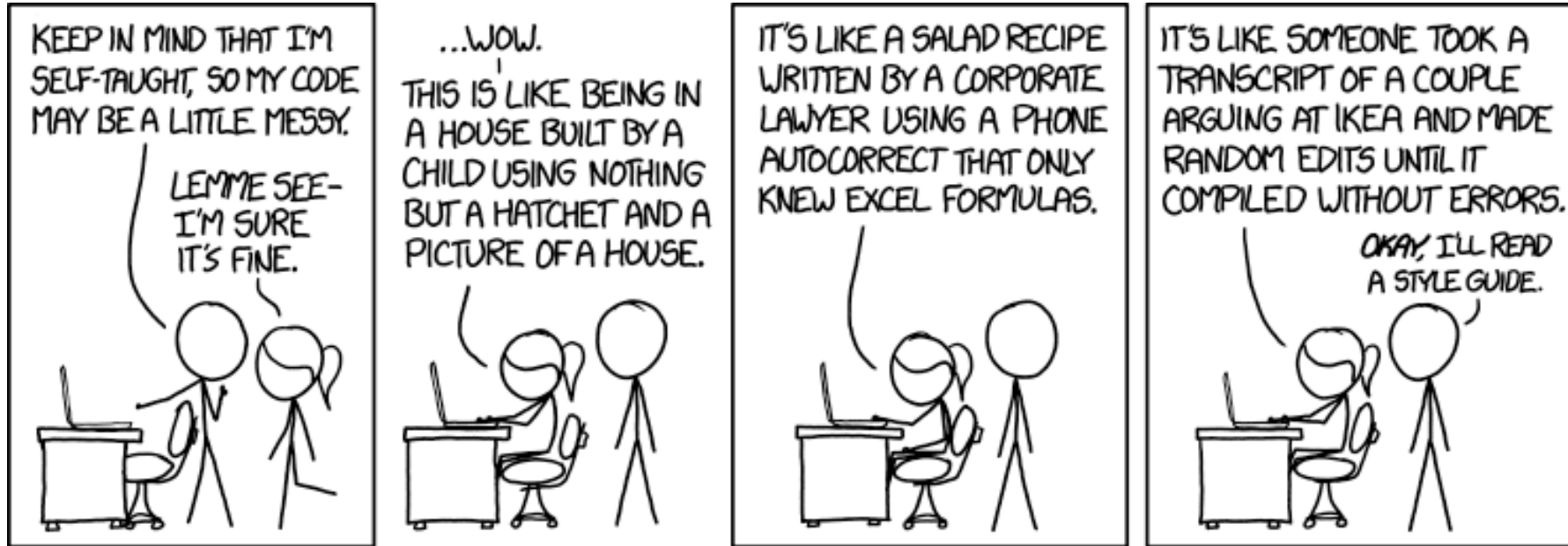
Leerdoelen week 3 les 2. Je leert hoe je:

- gebruik kunt maken van verschillende **C coding standards**;
- gebruik kunt maken van verschillende **static code analysis tools**, waaronder **Cppcheck**, om de kwaliteit van je C-code te verbeteren.



I hate reading
other people's code.

Style guide



Bron: <https://xkcd.com/1513/>

Wat is het? Welke coding standards zijn beschikbaar voor **C**?

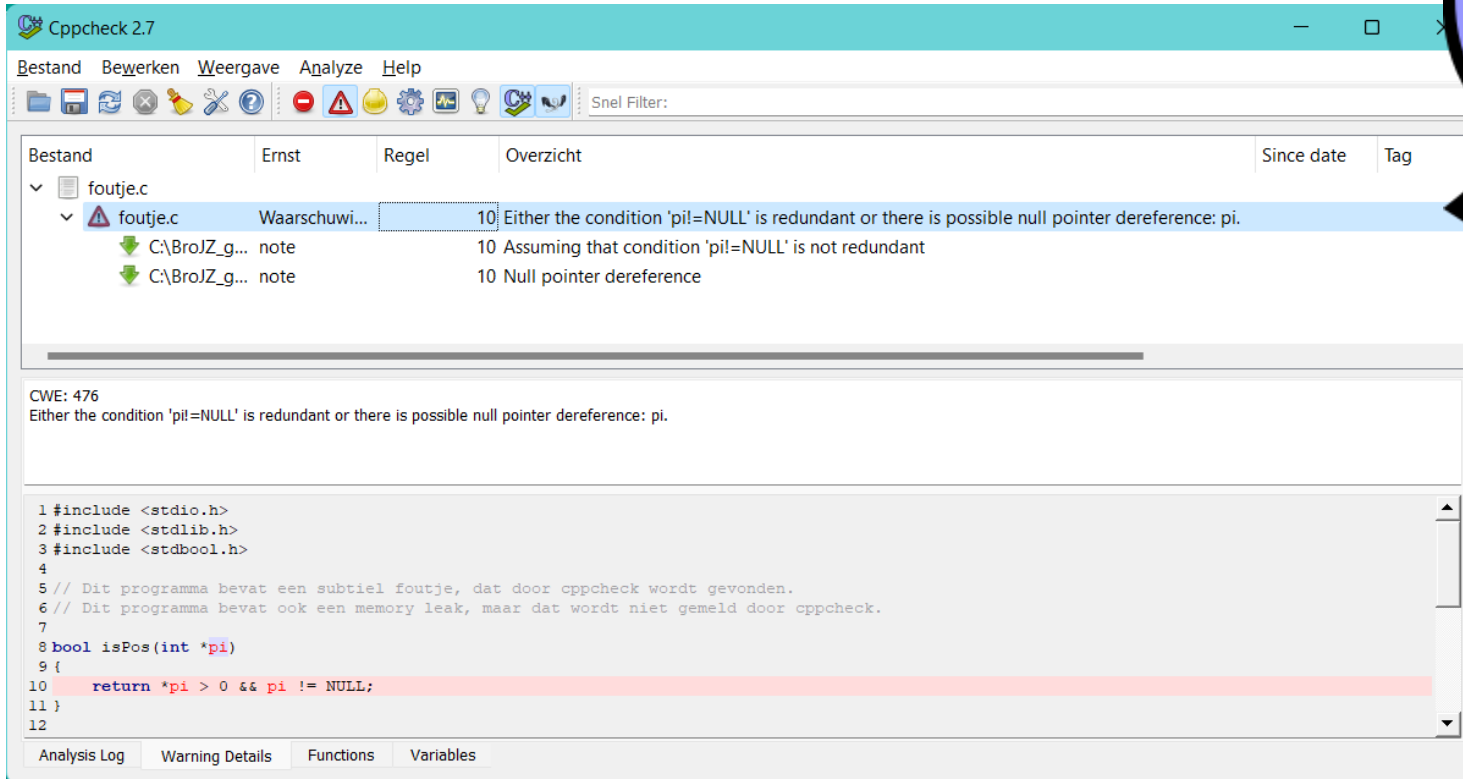
- MISRA (Motor Industry Software Reliability Association) [MISRA C:2023](#) (2023), betaald £15.
- Barr Group: [Embedded C Coding Standard](#) (2018), gratis: [klik hier!](#)
- SEI (Software Engineering Institute) CERT (Computer Emergency Response Team) [C Coding Standard](#) (2016), gratis: [klik hier!](#)
- IPA (Information-technology Promotion Agency, Japan) [Embedded System development Coding Reference guide](#) [C Language Edition] (2018), gratis: [klik hier!](#)
- ... vele anderen ...

Static code analysis

- Wat is het?
- Welke **gratis** tools zijn beschikbaar voor C?
 - **Cppcheck**
 - Lint, Splint, Adlint, OCLint, PC-Lint, ...
 - Clang –analyze
 - ...
- Welke **commerciële** tools zijn er voor C?
 - Coverty
 - Klocwork
 - Parasoft
 - Helix QAC
 - ...

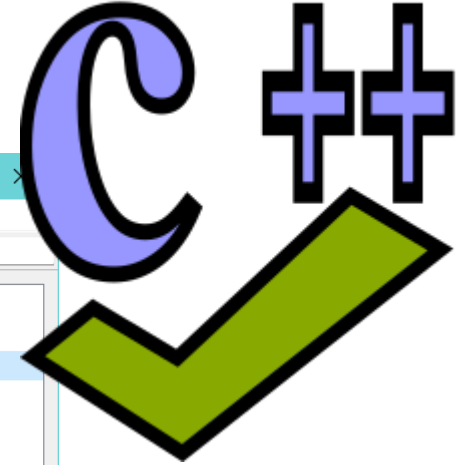


<http://cppcheck.sourceforge.net/>



The screenshot shows the Cppcheck 2.7 application window. The menu bar includes Bestand, Bewerken, Weergave, Analyze, and Help. The toolbar contains various icons for file operations and analysis. The main window displays a tree view of files, with 'foutje.c' expanded to show a warning: '10: Either the condition 'pi!=NULL' is redundant or there is possible null pointer dereference: pi.' Below this, two notes are listed: '10 Assuming that condition 'pi!=NULL' is not redundant' and '10 Null pointer dereference'. The bottom pane shows the source code for 'foutje.c', with the line 'return *pi > 0 && pi != NULL;' highlighted in red. The code includes headers for <stdio.h>, <stdlib.h>, and <stdbool.h>. Comments indicate that the program contains a subtle bug found by cppcheck and a memory leak not reported by it.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <stdbool.h>
4
5 // Dit programma bevat een subtiel foutje, dat door cppcheck wordt gevonden.
6 // Dit programma bevat ook een memory leak, maar dat wordt niet gemeld door cppcheck.
7
8 bool isPos(int *pi)
9 {
10  return *pi > 0 && pi != NULL;
11 }
12
```



Voorbeeld foutje.c

https://bitbucket.org/HR_ELEKTRO/ems31/raw/master/Opdrachten/progs/cppcheck/foutje.c

```
// Bepaal of pointer pi naar een positief getal wijst
bool isPos(int *pi)
{
    return *pi > 0 && pi != NULL;
}

int main(void)
{
    int *p = malloc(sizeof(int));
    if (p != NULL)
        *p = 15;
    if (isPos(p))
        printf("*p is positief, zoals verwacht!\n");
}
```


Cppcheck in VS Code

Cppcheck Plug-in
A plug-in for Cppcheck, capable of checking folders or editor tabs, shows output in the output channel, severity options available.
NathanJ

Gcov Viewer 101ms
Decorate C/C++ source files with code coverage information generated by gcov.
Jacques Lucke

```
C foutje.c 2 X
Opdrachten > cppcheck > C foutje.c > ...
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <stdbool.h>
4
5 // Dit programma bevat een subtiel foutje, dat door cppcheck wordt gevonden.
6 // Dit programma bevat ook een memory leak, maar dat wordt niet gemeld door cppcheck.
7
8 // Bepaal of pointer pi naar een positief getal wijst
9 bool isPos(int *pi)
10 {
11     ... return *pi > 0 && pi != NULL;
12 }
```

File Explorer showing a tree structure with files like `foutje.c`, `makefile`, `linked_list`, `valgrindMemcheck`, `memcheck-output.txt`, `memcheck.c`, `stupid-output.txt`, `stupid.c`, `.bash_history`, `.bash_logout`, `.bash_profile`, `.bashrc`, `.gitconfig`, `.gitignore`, `.lesshst`, `.my-credentials`, and `bitbucket-pipelines.yml_old`. A context menu is open over `foutje.c` with options: Open to the Side (Ctrl+Enter), Open With..., Reveal in Explorer (Shift+Alt+R), Open in Integrated Terminal, Select for Compare, Open Timeline, Cut (Ctrl+X), Copy (Ctrl+C), Download..., Copy Path (Shift+Alt+C), Copy Relative Path (Ctrl+K Ctrl+Shift+C), Rename... (F2), Delete Permanently (Delete), and a custom button 'Check C/C++ file'.

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS

▼ C foutje.c ~/Opdrachten/cppcheck 3

- Null pointer dereference: pi CppCheck (c-cpp-flylint)(ctunullpointer) [Ln 11, Col 5]
- Either the condition 'pi!=NULL' is redundant or there is possible null pointer dereference: pi. CppCheck (c-cpp-flylint)(nullPointerRedundantCheck) [Ln 11, Col 5]
- Parameter 'pi' can be declared as pointer to const CppCheck (c-cpp-flylint)(constParameterPointer) [Ln 9, Col 1]

CCS kan MISRA-C 2004 checken

Properties for line_in_2_line_out

MISRA-C:2004

Configuration: Debug [Active] Manage Configurations...

Enable checking of MISRA-C:2004 rules (--check_misra)

- 1: Environment
- 2: Language extensions
- 3: Documentation
- 4: Character sets
- 5: Identifiers
 - 5.2 [Required]: Identifiers in an inner scope shall not use the same name as those in an outer scope.
 - 5.3 [Required]: A typedef name shall be a unique identifier.
 - 5.4 [Required]: A tag name shall be a unique identifier.
 - 5.6 [Advisory]: No identifier in one name space should have the same name as an identifier in another name space.
 - 5.7 [Advisory]: No identifier name should be reused.
- 6: Types
- 7: Constants
- 8: Declarations and definitions

Command: "all"

Set severity of MISRA 'advisory' rule class (--misra_advisory) warning

Set severity of MISRA 'required' rule class (--misra_required) error

Buttons: None, All, Required, Advisory, Expand All, Collapse All, Apply and Close, Cancel

CCS MISRA-C 2004 code check

The screenshot shows the CCS IDE interface. At the top, there is a menu bar with options: File, Edit, View, Navigate, Project, Run, Scripts, Window, Help. Below the menu bar is a toolbar with various icons for file operations, execution, and navigation. The main workspace area is titled "workspa".

The Problems window is open, showing a summary of 281 errors, 20 warnings, and 0 others. A filter is applied, matching 120 of 301 items. The window is divided into two tabs: "Problems" (selected) and "Advice".

The Problems window displays the following error messages:

- 281 errors, 20 warnings, 0 others (Filter matched 120 of 301 items)
- Description
- Errors (100 of 281 items)
 - #1376-D (MISRA-C:2004 1.1/R) Ensure strict ANSI C mode (-ps) is enabled
 - #1376-D (MISRA-C:2004 1.1/R) Ensure strict ANSI C mode (-ps) is enabled
 - #1420-D (MISRA-C:2004 16.5/R) Functions with no parameters shall be declared and defined with
 - #1445-D (MISRA-C:2004 20.9/R) The input/output library <stdio.h> shall not be used in productio
 - #1455-D (MISRA-C:2004 2.2/R) Source code shall only use /* ... */ style comments

Volgende week...

Werken aan [Eindopdracht 1](#)

```
> hallo
Onbekend commando: "hallo"
> nul
leds: groen = 0, geel = 0, rood = 0
> zes
leds: groen = 1, geel = 1, rood = 0
> schakelroodaan
leds: groen = 1, geel = 1, rood = 1
> schakelgroenuit
leds: groen = 0, geel = 1, rood = 1
> schakelgeelom
leds: groen = 0, geel = 0, rood = 1
> schakelgeelom
leds: groen = 0, geel = 1, rood = 1
> □
```

Aan de slag!

Aan de slag met [Opdrachten_Week_3.6.pdf](#)

