MONASH University

MONASH
ENGINEERING
ENG1060

DEBUGGING

Presented by Tony Vo

Slides by Tony Vo

GROUP OF EIGHT AUSTRALIA

DEBUGGING

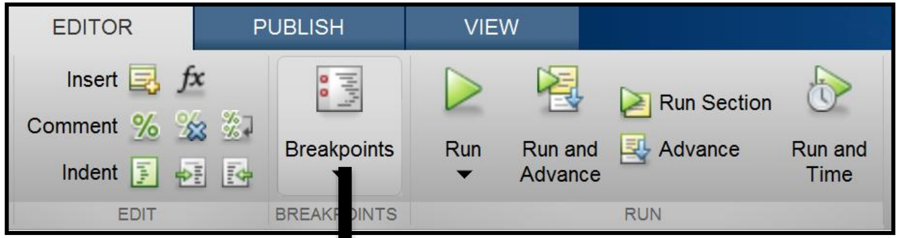
- Debugging is the task of identifying and correcting errors (bugs) in code
- Sometimes, identifying a bug is much more time consuming and challenging than writing the code itself
  - Especially true for poorly written code
- Good programming practices will generally reduce the likelihood of bugs
  - Functions
  - Comments
  - Indenting
  - Pseudocode
  - Descriptive variable names

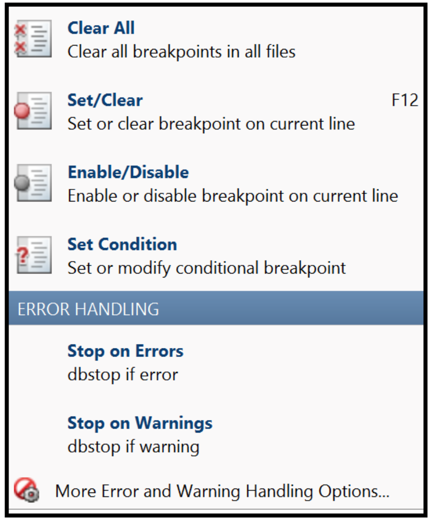
MATLAB DEBUGGER

- MATLAB has a debugger tool to assist with debugging
  - Allows for code stops at a certain point
  - Allows for "stepping" through code
  - Especially useful in complicated loops and functions

MATLAB DEBUGGER: BREAKPOINTS

- Breakpoints represent a "stop sign" for code execution
  - Valid breakpoints appear as red circles
  - Grey circles indicate a syntax error or unsaved .m file
  - Conditional breakpoints appear as yellow circles




Clear All
Clear all breakpoints in all files

Set/Clear F12
Set or clear breakpoint on current line

Enable/Disable
Enable or disable breakpoint on current line

Set Condition
Set or modify conditional breakpoint

ERROR HANDLING

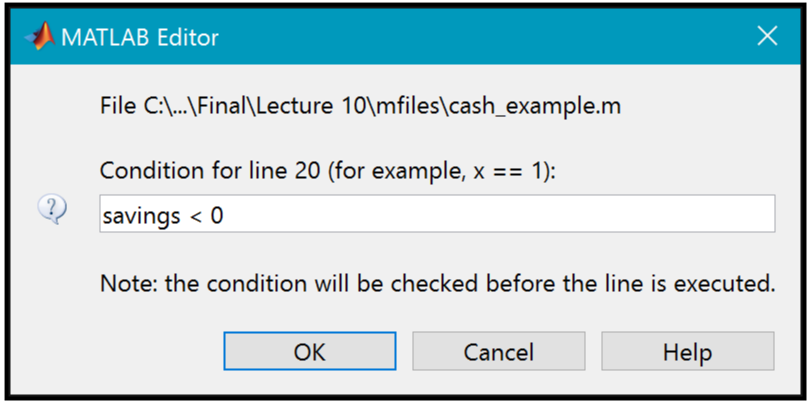
Stop on Errors
dbstop if error

Stop on Warnings
dbstop if warning

More Error and Warning Handling Options...

MATLAB DEBUGGER: BREAKPOINTS

- When setting conditional breakpoints a dialog box will appear
  – This breakpoint will only activate if the condition is true



File C:\...\Final\Lecture 10\mfiles\cash_example.m

Condition for line 20 (for example, x == 1):

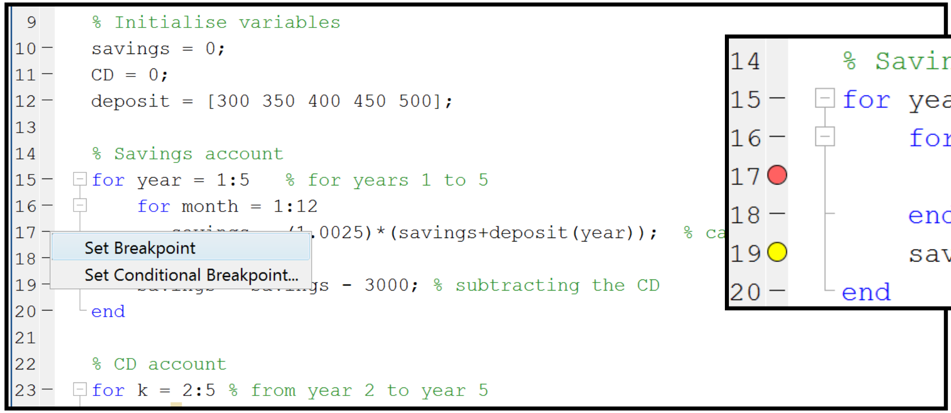
```
savings < 0
```

Note: the condition will be checked before the line is executed.

SETTING BREAKPOINTS

- Clicking next to the line number will allow you to set breakpoints too
  - Left click will set an unconditional breakpoint
  - Right click will give both breakpoint types

- Conditional breakpoints: only stops the code if the condition is met

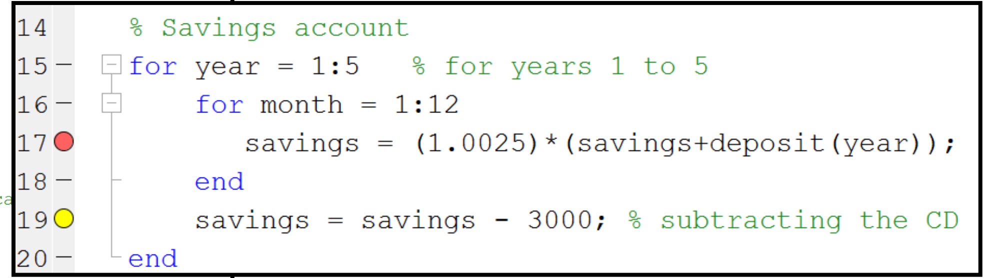


```
% Initialise variables
savings = 0;
CD = 0;
deposit = [300 350 400 450 500];

% Savings account
for year = 1:5   % for years 1 to 5
    for month = 1:12
        savings = (1.0025)*(savings+deposit(year));   % ca

    savings = savings - 3000; % subtracting the CD
end

% CD account
for k = 2:5 % from year 2 to year 5
```

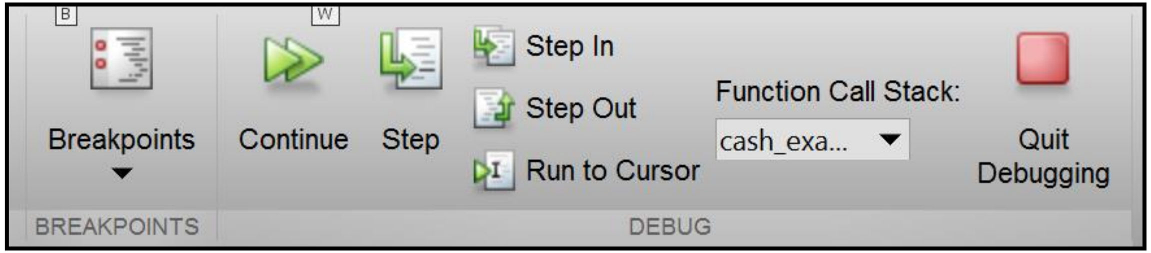


```
% Savings account
for year = 1:5   % for years 1 to 5
    for month = 1:12
        savings = (1.0025)*(savings+deposit(year));
    end
    savings = savings - 3000; % subtracting the CD
end
```

ENG1060

RUNNING WITH BREAKPOINTS

- Once your code hits a break point, several options will appear
  - Continue: Continues running the code until next breakpoint
  - Step: Steps through a single line
  - Step in/out: Steps in and out of function files and loops
  - Run to cursor: Runs the code to the current cursor position
  - Quit debugging: Exits debugging mode



SUMMARY

- Debugging code can be a time consuming process
- Ensure you adopt good programming practices to decrease the likelihood of bugs
- Use the debugger tool to step through complicated code line-by-line

- Do conditional breakpoints check the condition before or after running the line it's set at?