Yale-NUS College - 2018-2019

Bruno Bodin

YSC2227: INTRO TO C

week 04.1.debug (auto-generated)

WHAT IS YOUR FAVORITE BUG?

```
> ./a.out
Segmentation fault (core dumped)
```

GDB: GNU DEBUGGER

- Executes programs step by step
- Shows variable addresses and values
- Can also expose the current state of the stack

USING GDB - GET THE BACKTRACE

- Your program needs to be compiled with the `-g` flag! Otherwise …

```
(gdb) file a.out
Reading symbols from a.out...(no debugging symbols found)...done.
(gdb) quit
```

- Once you do `gcc -g` :

```
> gdb
(gdb) file a.out
Reading symbols from a.out...done.
(gdb) run
Starting program: /tmp/a.out

Program received signal SIGSEGV, Segmentation fault.
0x00000000004004a6 in main () at test.c:4
4   *ptr = 0;

(gdb) bt
```

USING GDB - BREAK POINTS AND PRINT

- You can stop the program within its execution:

```
> gdb a.out
(gdb) break test.c:4
(gdb) break function1
(gdb) run
Breakpoint 1, main () at test.c:4
4    *ptr = 0;
(gdb) print ptr
$1 = (int *) 0x400496 <main>...
(gdb) print &ptr
```

- Once a break happens you can see variables values:

```
(gdb) print ptr
$1 = (int *) 0x400496 <main>...
(gdb) print &ptr
...
```

- There is also conditionnal break :

```
(gdb) break test.c:4 if ptr == 0
```

USING GDB - RUN STEP BY STEP

```
> gdb a.out
(gdb) break main
(gdb) run
Breakpoint 1, main () at test.c:3
(gdb) step
```

USING GDB - WATCH

```
(gdb) break main
(gdb) run
(gdb) watch ptr
Hardware watchpoint 2: ptr
(gdb) next
Hardware watchpoint 2: ptr

Old value = (int *) 0x0
New value = (int *) 0x400496 <main>
main () at test.c:4
4   *ptr = 0;
```

TOPICS COVERED

- Variables and Assignement Operators ✓ (T. Bailey, Chapter 1 and 2)
- Numeric Data Types and Conversion ✓ (T. Bailey, Chapter 2)
- Arrays ✓ (T. Bailey, Chapter 8)
- Arithhmetic and bitwise operators ✓ (T. Bailey, Chapter 2 and 12)
- Compilation, flags, and command-line arguments ✓ (D. Harris C.10)
- Pointers ✓ (T. Bailey, Chapter 7)
- C functions ✓ (T. Bailey, Chapter 4)
- Files and I/O ✓ (T. Bailey, Chapter 13)
- Control structures, logic operators, and loops ✓ (T. Bailey, Chapter 3)
- Scope ✓ (T. Bailey, Chapter 5)
- Structures and Unions ✓ (T. Bailey, Chapter 11 and 14)
- Memory management and segmentation ✓ (T. Bailey, Chapter 9)
- Basic libraries
- Makefile
- Debugging ✓ (`https://www.cs.cmu.edu/~gilpin/tutorial/`)

KEY POINTS

REFERENCES

- cook and magician:
  https://pixabay.com/en/users/graphicmama-team-2641041/