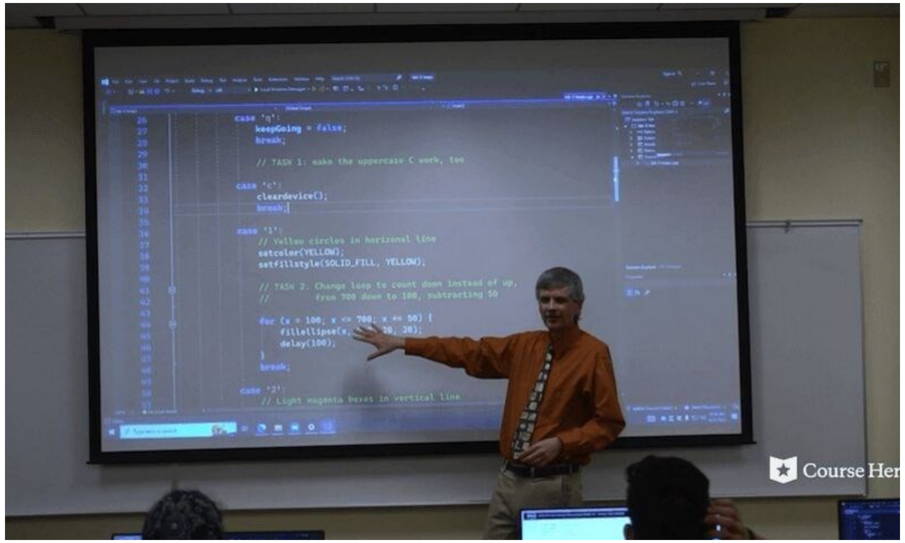
5118020-03 Operating Systems

Homework 1. Autojudge

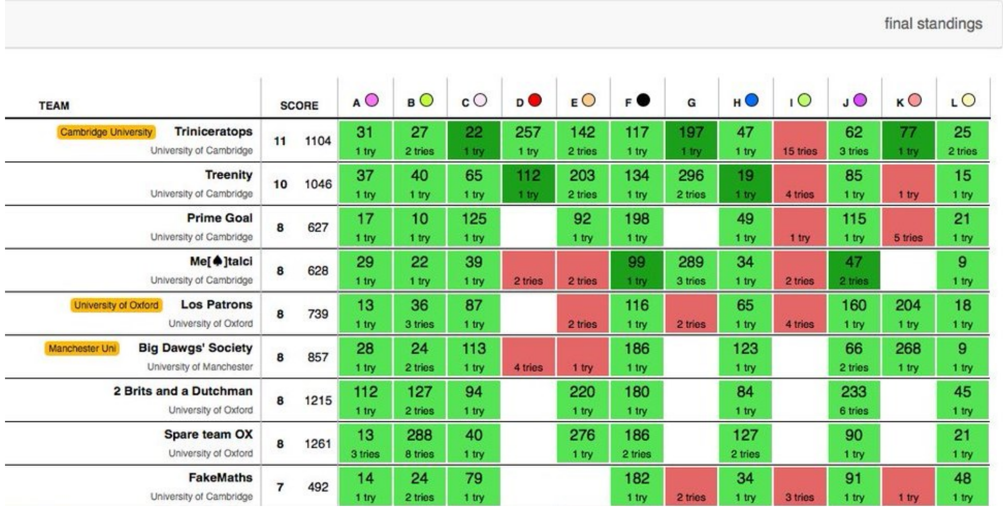
- Revision 1 (Apr 5, 2024)

Shin Hong

Scenario



```
case 'q':
    keepGoing = false;
    break;

    // TASK 1: make the uppercase C work, too

case 'c':
    cleardevice();
    break;

case '1':
    // Yellow circles in horizonal line
    setcolor(YELLOW);
    setfillstyle(SOLID_FILL, YELLOW);

    // TASK 2: Change loop to count down instead of up,
    //         from 700 down to 100, subtracting 50

    for (x = 100; x <= 700; x += 50) {
        fillellipse(x,      30, 30);
        delay(100);
    }
    break;

case '2':
    // Light magenta boxes in vertical line
```



final standings

TEAM	SCORE		A ●	B ●	C ○	D ●	E ●	F ●	G	H ●	I ●	J ●	K ●	L ○
Cambridge University Triniceratops University of Cambridge	11	1104	31 1 try	27 2 tries	22 1 try	257 1 try	142 2 tries	117 1 try	197 1 try	47 1 try	15 tries	62 3 tries	77 1 try	25 2 tries
Treenity University of Cambridge	10	1046	37 1 try	40 1 try	65 1 try	112 1 try	203 2 tries	134 1 try	296 2 tries	19 1 try	4 tries	85 1 try	1 try	15 1 try
Prime Goal University of Cambridge	8	627	17 1 try	10 1 try	125 1 try		92 1 try	198 1 try		49 1 try	1 try	115 1 try	5 tries	21 1 try
Me[♠]talci University of Cambridge	8	628	29 1 try	22 1 try	39 1 try	2 tries	2 tries	99 1 try	289 3 tries	34 1 try	2 tries	47 2 tries		9 1 try
University of Oxford Los Patrons University of Oxford	8	739	13 1 try	36 3 tries	87 1 try		2 tries	116 1 try	2 tries	65 1 try	4 tries	160 1 try	204 1 try	18 1 try
Manchester Uni Big Dawgs' Society University of Manchester	8	857	28 1 try	24 2 tries	113 1 try	4 tries	1 try	186 1 try		123 1 try		66 2 tries	268 1 try	9 1 try
2 Brits and a Dutchman University of Oxford	8	1215	112 1 try	127 2 tries	94 1 try		220 1 try	180 1 try		84 1 try		233 6 tries		45 1 try
Spare team OX University of Oxford	8	1261	13 3 tries	288 8 tries	40 1 try		276 1 try	186 2 tries		127 2 tries		90 1 try		21 1 try
FakeMaths University of Cambridge	7	492	14 1 try	24 2 tries	79 1 try			182 1 try	2 tries	34 1 try	3 tries	91 1 try	1 try	48 1 try

Homework 1. Autojudge

5118020-03
Operating Systems

2024-04-05

Overview

- Construct *autojudge*, a C program that builds another C program, run tests on it, and reports the results to the user
  - a target program is an arbitrary C program (e.g., student’s submission)
  - test inputs and outputs (answers) are given

- Point of study
  - process creation and control: fork, exec, wait
  - inter-process communication: signal, unnamed pipe

- Timelines
  - Apr 2: First announcement & team arrangement
  - Apr 5: Second announcement (as video)
  - Apr 10-12: Help desks
  - Apr 17, 9 PM: Artifact submission deadline (source code)
  - Apr 18, 9 PM: Presentation submission deadline (video record)

Autojudge (1/3)

- Input: receive two directories, one with input files and the other with answer files as input
  - an input file and the corresponding answer file (i.e., the expected output) have the same file name
  - command-line interface

```
$ ./autojudge –i <inputdir> -a <outputdir> -t <timelimit> <target src>
```

- Process
  (1) check if given command-line arguments are valid
  (2) compile the target program and generate the executable
  (3) for each input file, run the target program with it and check if the result is identical to corresponding answer
  (4) report the overall results to the user

Autojudge (2/3)

- Target program
  - Receive input from the standard input
  - Generate output to the standard output
  - Return 0 when the program terminates without errors

- Judge criteria
  - **Compile Error**
  - **Timeout**: an execution exceeds a given time limit
  - **Runtime Error**: the program crashes or returns a non-zero exit code
  - **Wrong Answer (n/m)**: among n test inputs, only with m inputs, the program generates the outputs identical to the answers
  - **Correct**: successfully complied & passes with all test inputs

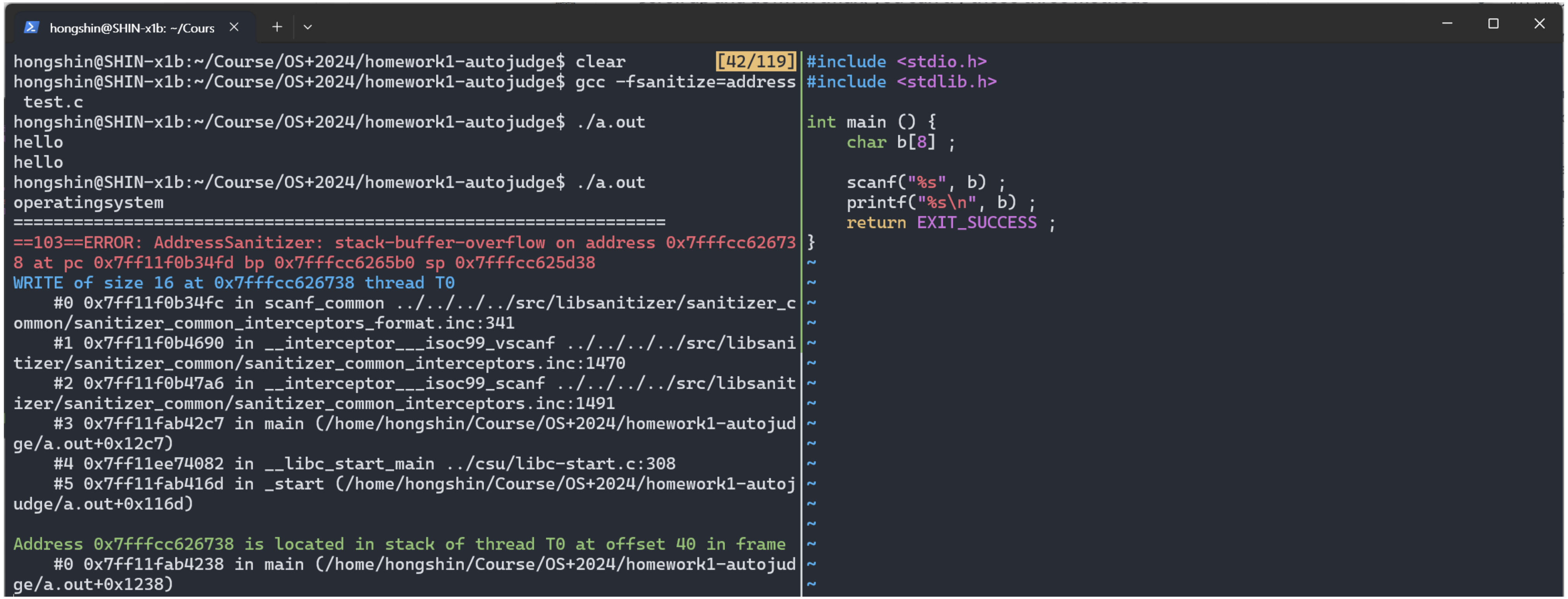
Autojudge (3/3)

- Output
  - If the judgement is **Compile Error**, print the compile error message
  - If the judgement is **Timeout** or **Wrong Answer**, print the number of tests where the target program produced the correct output, the timeout, and the wrong output, respectively.
  - If the judgement is **Wrong Answer**, print the number of tests that the target program generated the correct outputs.
  - If the judgement is **Correct**, print out the sum of the running time of all test executions (in milliseconds)

Compile

- Use GCC with sanitizer
  - `gcc -fsanitize=address <target src>`

c.f., https://www.osc.edu/resources/getting_started/howto/howto_use_address_sanitizer



```
hongshin@SHIN-x1b:~/Course/OS+2024/homework1-autojudge$ clear           [42/119]
hongshin@SHIN-x1b:~/Course/OS+2024/homework1-autojudge$ gcc -fsanitize=address
 test.c
hongshin@SHIN-x1b:~/Course/OS+2024/homework1-autojudge$ ./a.out
hello
hello
hongshin@SHIN-x1b:~/Course/OS+2024/homework1-autojudge$ ./a.out
operatingsystem
=================================================================
==103==ERROR: AddressSanitizer: stack-buffer-overflow on address 0x7fffcc62673
8 at pc 0x7ff11f0b34fd bp 0x7fffcc6265b0 sp 0x7fffcc625d38
WRITE of size 16 at 0x7fffcc626738 thread T0
    #0 0x7ff11f0b34fc in scanf_common ../../../../src/libsanitizer/sanitizer_c
ommon/sanitizer_common_interceptors_format.inc:341
    #1 0x7ff11f0b4690 in __interceptor___isoc99_vscanf ../../../../src/libsani
tizer/sanitizer_common/sanitizer_common_interceptors.inc:1470
    #2 0x7ff11f0b47a6 in __interceptor___isoc99_scanf ../../../../src/libsanit
izer/sanitizer_common/sanitizer_common_interceptors.inc:1491
    #3 0x7ff11fab42c7 in main (/home/hongshin/Course/OS+2024/homework1-autojud
ge/a.out+0x12c7)
    #4 0x7ff11ee74082 in __libc_start_main ../csu/libc-start.c:308
    #5 0x7ff11fab416d in _start (/home/hongshin/Course/OS+2024/homework1-autoj
udge/a.out+0x116d)

Address 0x7fffcc626738 is located in stack of thread T0 at offset 40 in frame
    #0 0x7ff11fab4238 in main (/home/hongshin/Course/OS+2024/homework1-autojud
ge/a.out+0x1238)
```

```
#include <stdio.h>
#include <stdlib.h>

int main () {
    char b[8] ;

    scanf("%s", b) ;
    printf("%s\n", b) ;
    return EXIT_SUCCESS ;
}
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

Assumptions & Requirements

- Assume that no more than 20 test inputs are given
- Assume that the target program is not malicious and does not bring any harmful side-effects

- Must use `getopt` to receive command-line arguments and use `gettimeofday` to obtain the current time
- Must check validity of command-line arguments and print out proper error messages when wrong arguments are given
- Must properly use the following system APIs:
  - `fork, pipe,  dup2, wait`
- Must not use the following system APIs:
  - `system, popen`

Submission

- All results must be submitted via LMS
  - Source code files
    - Submit all source code
    - You must provide a build script (e.g., bash script or Makefile) and its instruction document (e.g., README) if needed
  - Presentation
    - Submit the video record file; or, you can submit the URL to the presentation video on web

- No late submissions will be accepted

Video Presentation

- Take a 5-min video for reviewing the source code and testing the program
  - either in Korean or in English
  - every team member must take a part in presentation

- Your video must show the followings:
  - cases of the five different results
    - compile error, runtime error, timeout, wrong answers, correct answer
  - how your program detects and handles timeout cases

Notes

- Welcome your questions anytime on the Slack channel

- The team members must share the same responsibilities and take in charge of all tasks together
  - Peer evaluation follows immediately after the submission deadline
  - Inform me quickly if you keep fail to contact with your teammate

- It is strictly permitted to use auto-programming tools in any form