CS1010 Tutorial 4
Group BC1A

17 September 2020

Topics for today

Tracing.

Objectives

- Recap on Topics (Loop Invariant, Call Stack)
- Going through problem set 12, 13
- Common issues with Assignment 1 and 2
- Going through Exercise 1 and 2
- Summary, Q&A

Reminders (Mid terms)

- Date: *28 September, 2020 (Monday)*
- Time: 12 noon to 2pm
- Duration: **60 minutes**
- Online via Luminus Quiz
- 10% of your grade
- **Scope:**
  - Units 1-12
  - Tutorials 1-4
  - Assignments 1-2
- Format: **MCQs and Short Structured Questions**
- **Open book** (printed/written notes only)

Important details

- https://mysoc.nus.edu.sg/academic/e-exam-sop-for-students/
- https://nus-cs1010.github.io/2021-s1/slides/midterm.md

**If you need to take the online midterm on campus, please fill up the LumiNUS Survey on "Midterm Venue" before Thursday 23:59.*

Reminders (PE 1)

- Date: *3 October 2020 (Saturday)*
- Time: 9 am to 12noon
- Duration: **2 hours 30 minutes**
- Online via ssh into restricted PE hosts
- 10% of your grade
- **Scope**: same as midterm
- **Format**: Five programming problems
- **Open Book** (printed/written notes only)
- Calculator allowed (but not needed)
- Criteria: *correctness, style, and efficiency*

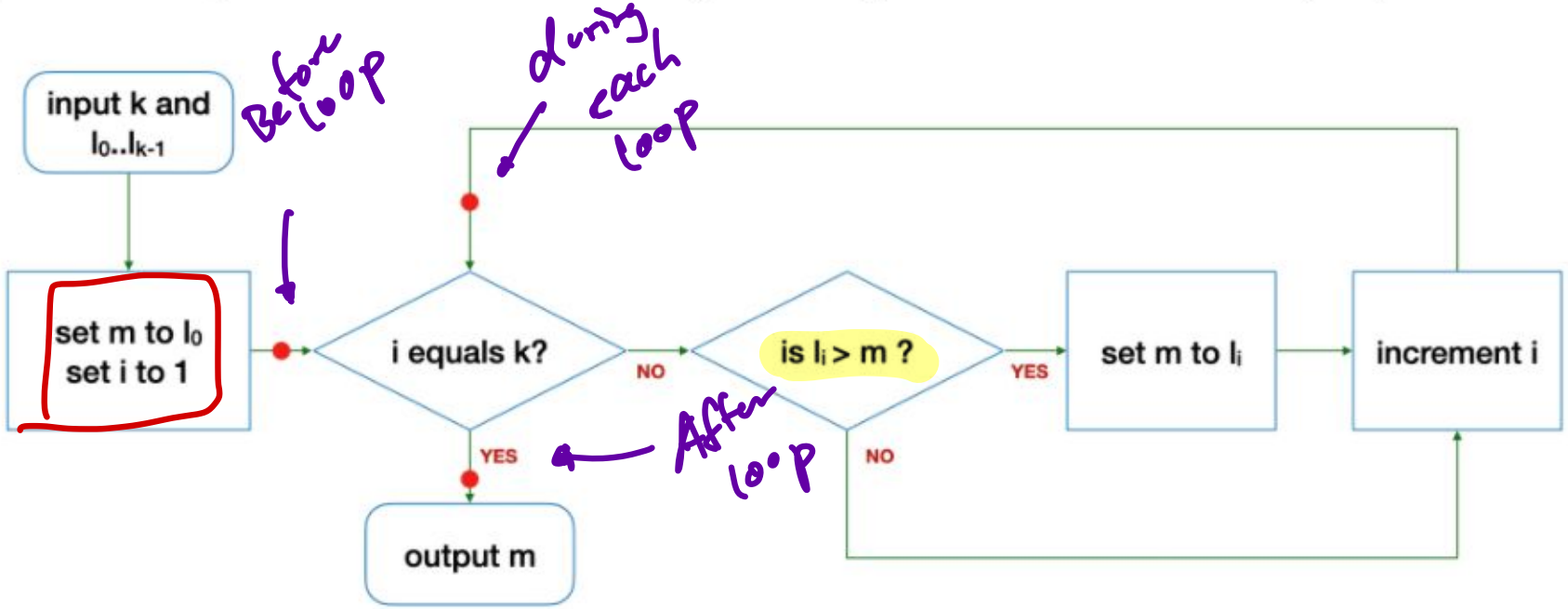
Important details

- https://mysoc.nus.edu.sg/academic/e-exam-sop-for-students/
- https://nus-cs1010.github.io/2021-s1/slides/pe1.md

loop Invariant

Problem 12.1(a) Question

(a) Consider the algorithm to find the maximum among a list of integers L with at least one element ($k > 0$) below:



The loop invariant for this loop must hold at the three points marked with the red dots: before the loop, after each iteration of the loop, and after the loop.

State the loop invariant, explain why it holds at the three points above, and threfore argue that the loop above correctly finds the maximum among the elements of the list L.

Problem 12.1(a) Solution

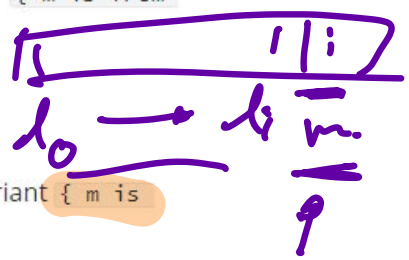
- The first step is to write the loop invariant. Since we are trying to find the max from a list, we want the following assertion `{ m is from L and m >= [l0... l{k-1}] }` to be true at the end of the loop.
- As we loop through, we need m to be the max among those we examined. So we want `{ m is from L and m >= [l0... l{i-1}] }`
- Check that it is true before we enter the loop.
  - `m = l0` and `i is 1`, so `{ m is from L and m >= [l0] }` is true.
  - Assuming that as we enter the loop (the no branch of "`i equals k`"), this loop invariant `{ m is from L and m >= [l0... l{i-1}] }` is true.
  - If `m >= l_i`, then we have `{ m is from L and m >= [l0 .. li] }`
  - If `m < l_i`, then we set `{ l_i > [l0 .. l{i-1}] }`, and after setting m to `l_i`, we have `{ m is from L and m >= [l0 .. li] }` as well.
  - Either way, we increase `i`, and now we have back to `{ m is from L and m >= [l0... l{i-1}] }`.
- The loop invariant is therefore true after each iteration.
- if `i equals k`, then the loop exit.
- { m is from L and m >= [l0... l{i-1}] } must be true as well since i and m` do not change.

So we have just shown that { m is from L and m >= [l0... l{i-1}] } is the loop invariant.

Now, we can argue why the algorithm is correct.

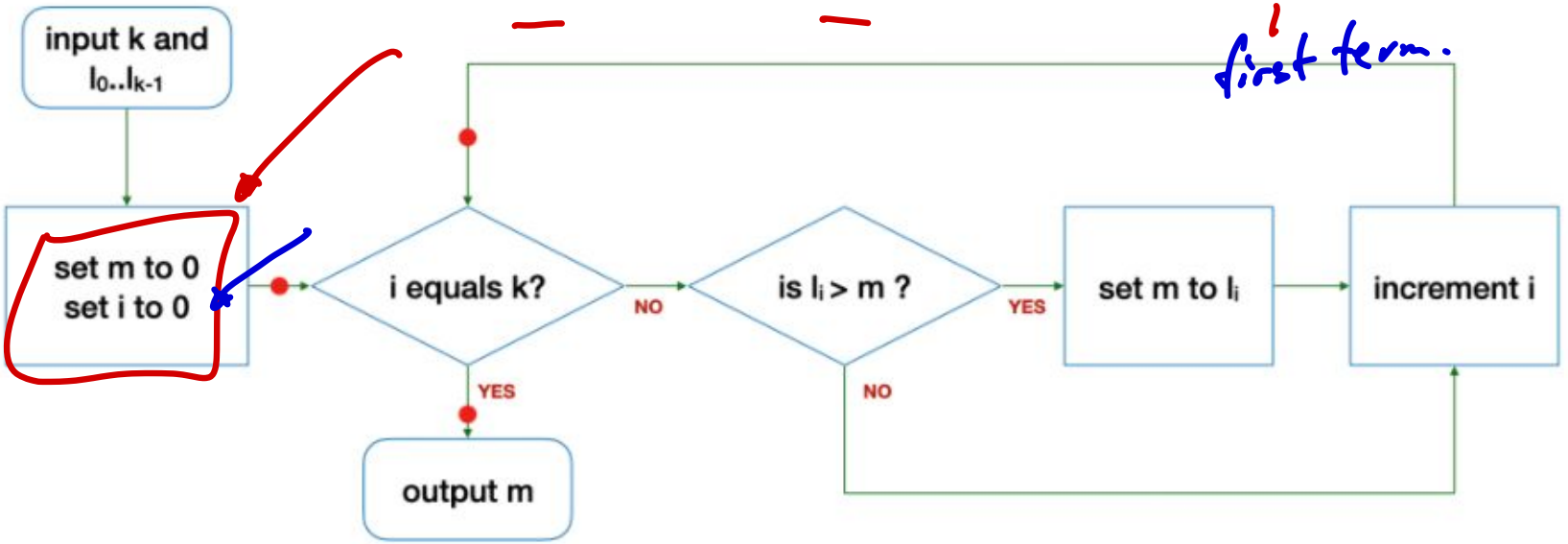
The assertion `{ i == k }` must be true when we exit the loop. So `{ m is from L and m >= [l0... l{k-1}] }`.

QED.



Problem 12.1(b) Question

(b) Now, consider a slightly different algorithm to find the maximum among a list of integers L with at least one element ($k > 0$) below:



Explain why you cannot find a loop invariant similar to Part (a) above, and therefore show that the algorithm does not correctly find the maximum in certain cases.

Problem 12.1(b) Solution

Let's consider the same invariant as above,

Check that it is true before we enter the loop.

`m = 0` and `i is 0`, so neither

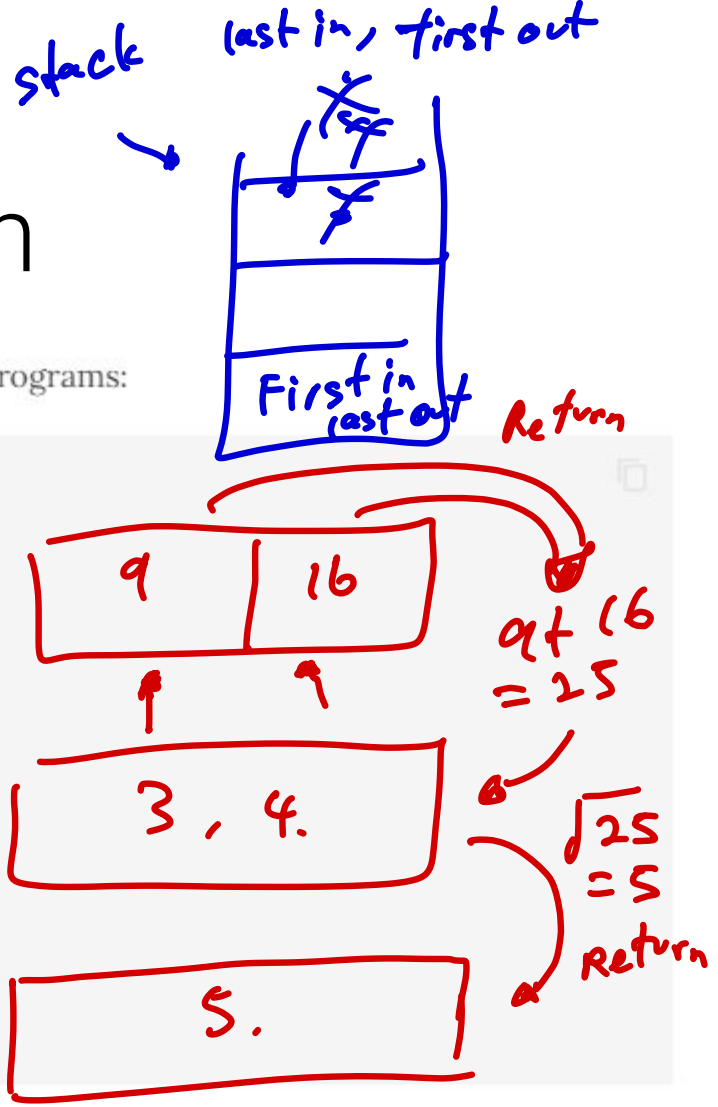
- `m` is from `L`, nor
- `m >= [10]`

is true

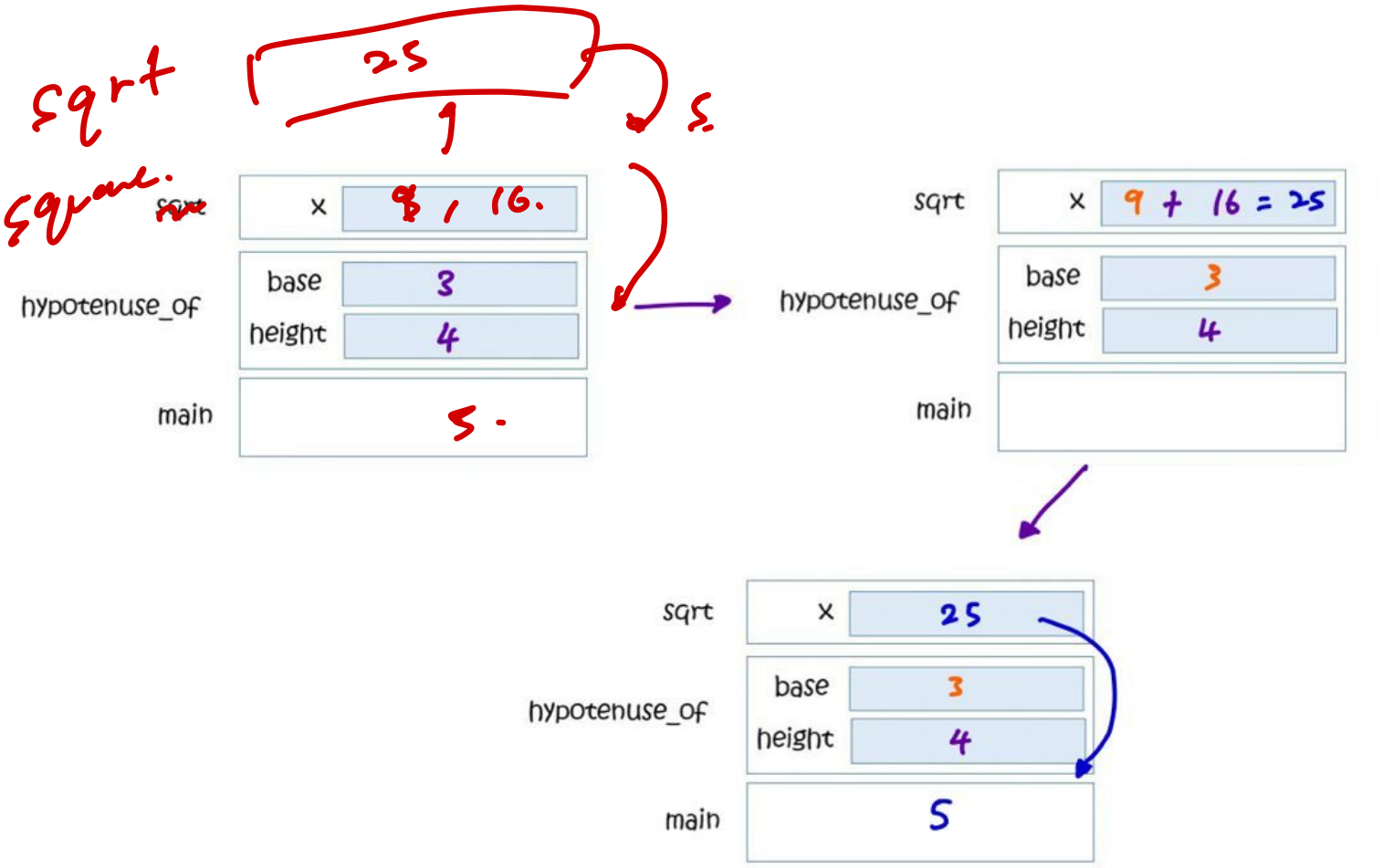
Problem 13.1 Question

Trace through what gets stored in the call stack when we run the following programs:

```
1  #include <math.h>
2
3  long square(long x)
4  {
5    return x*x;
6  }
7
8  double hypotenuse_of(long base, long height)
9  {
10   return sqrt(square(base) + square(height));
11 }
12
13 int main()
14 {
15   hypotenuse_of(3, 4);
16 }
```



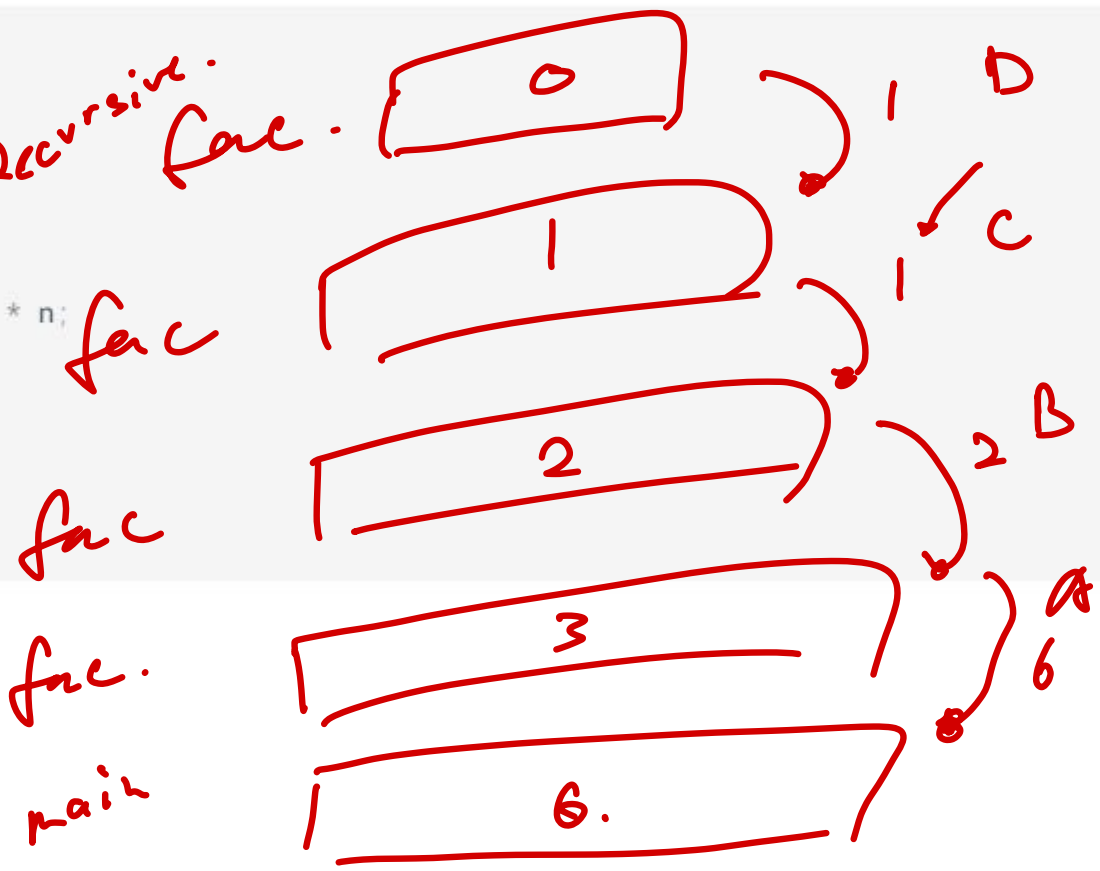
Problem 13.1 Solution



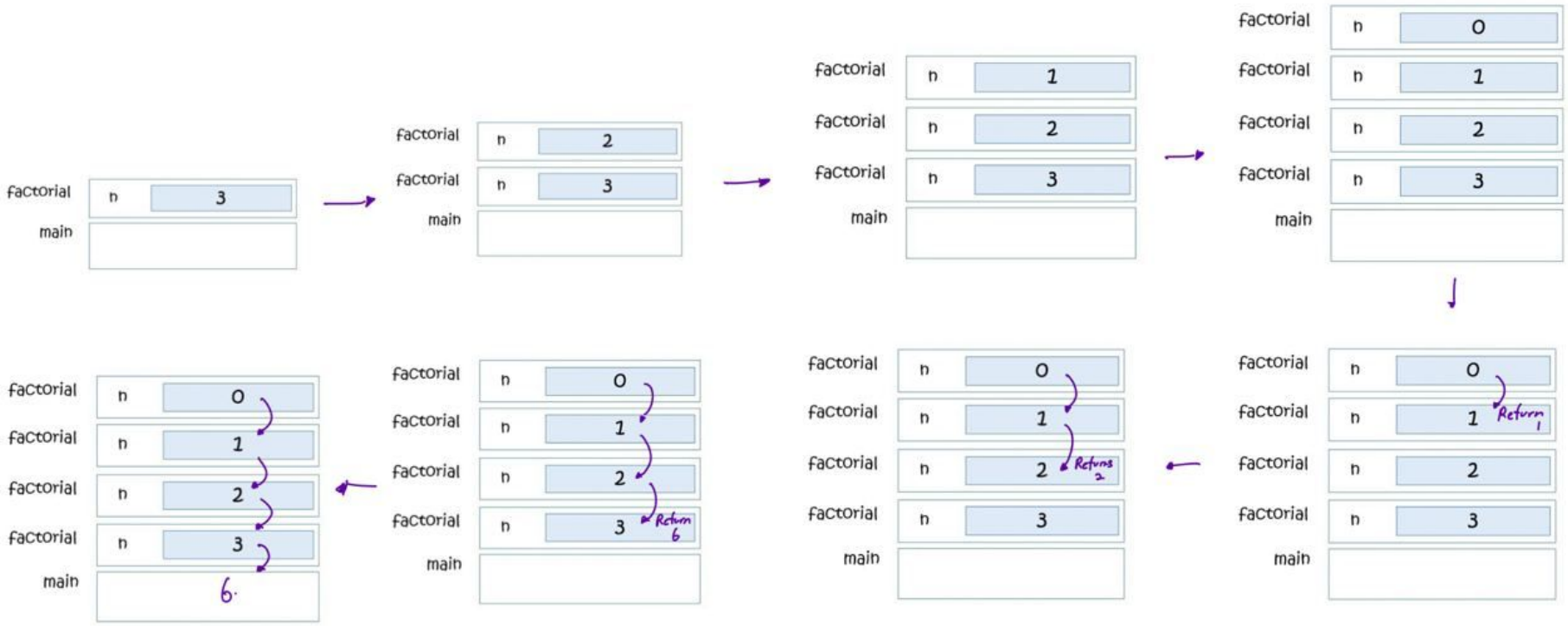
Problem 13.2 Question

Trace through what gets stored in the call stack when we run the following programs:

```
1  #include "cs1010.h"
2
3  long factorial(long n)
4  {
5    if (n == 0) {
6      return 1;
7    }
8    return factorial(n-1) * n;
9  }
10
11 int main()
12 {
13   factorial(3);
14 }
```



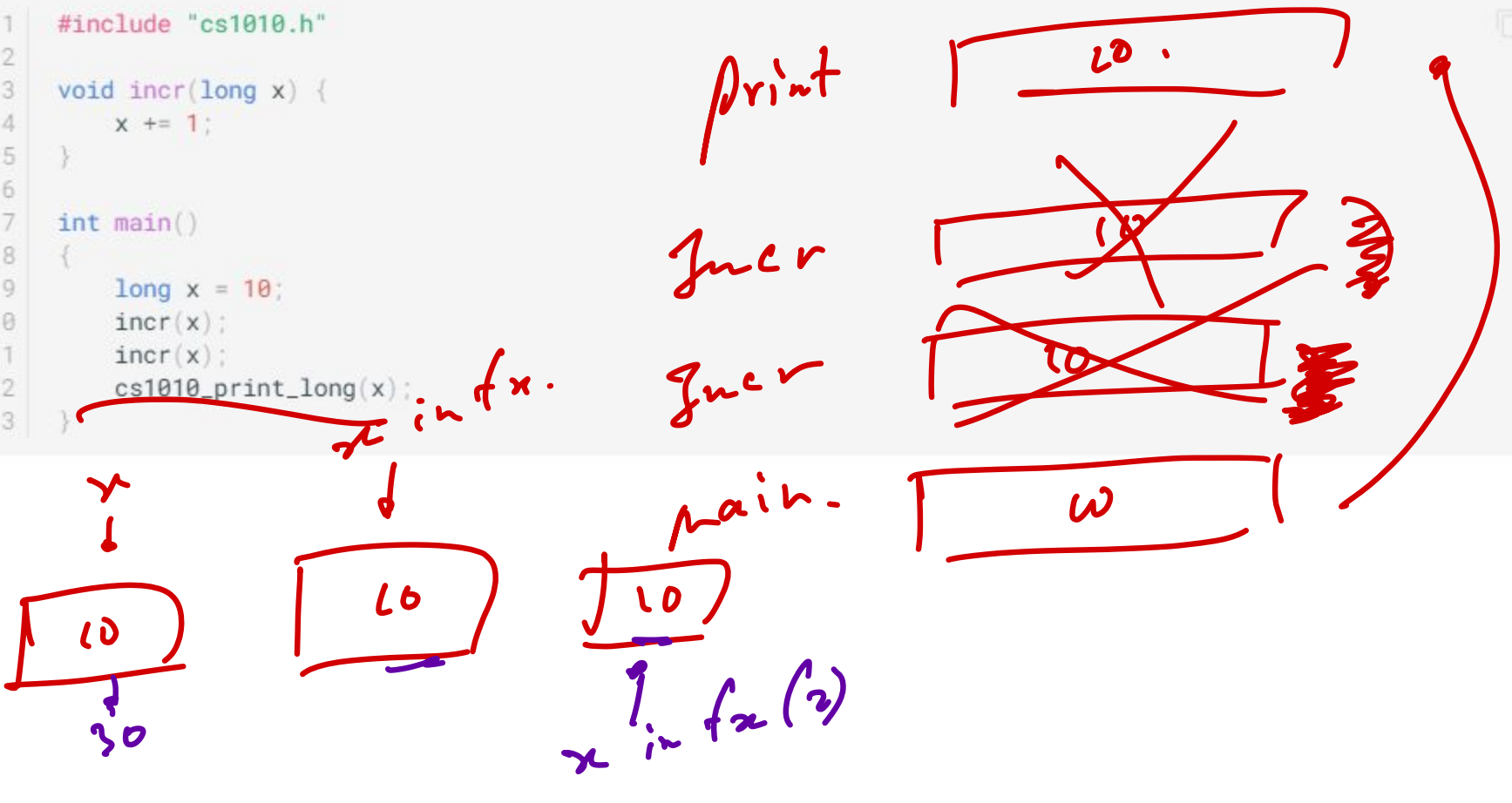
Problem 13.2 Solution



Problem 13.3 Question & Solution

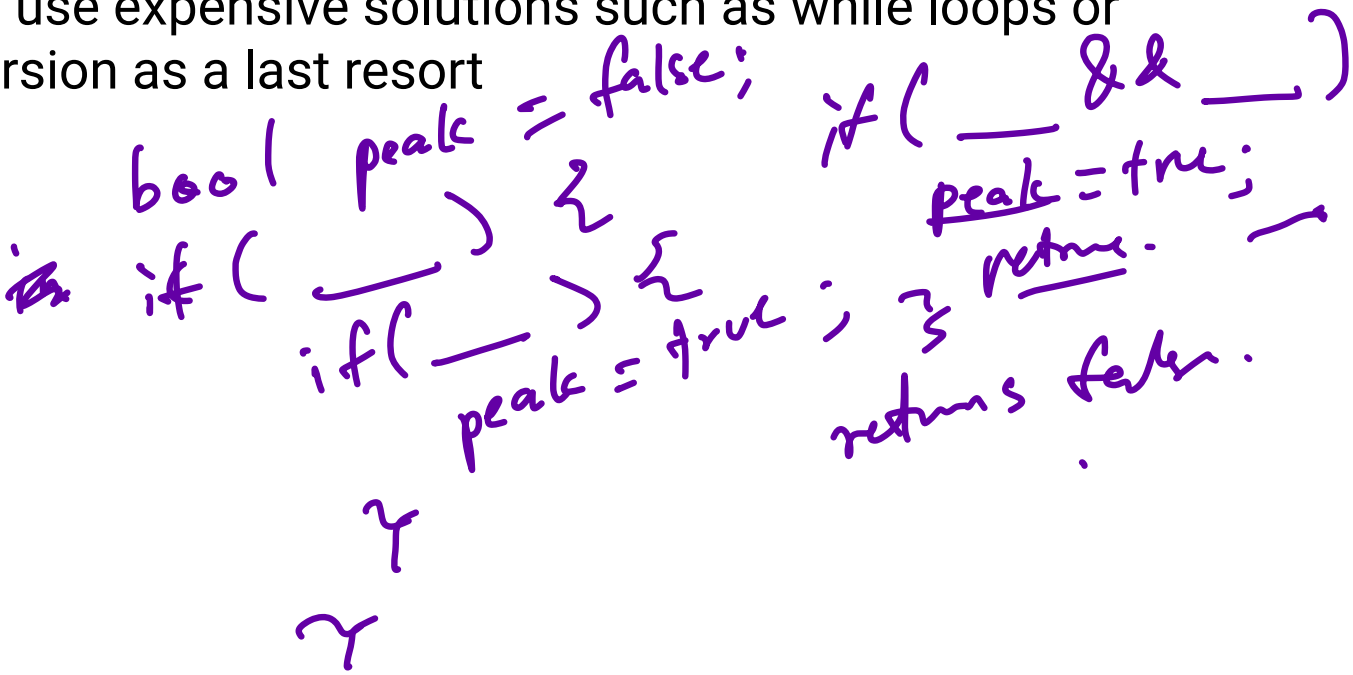
What will be printed by the program below? Trace through what gets stored in the call stack when we run the following programs:

```
1  #include "cs1010.h"
2
3  void incr(long x) {
4      x += 1;
5  }
6
7  int main()
8  {
9      long x = 10;
10     incr(x);
11     incr(x);
12     cs1010_print_long(x);
13 }
```



Assignment 1 Issues

- Take note of formatting
  - While formatting is not graded this assignment, it might be graded for the rest of the assignment
- Take note of redundant else statements
- See if there is a simpler logic to solve a question
  - Only use expensive solutions such as while loops or recursion as a last resort



Process of program writing

- Important steps for writing a program
  - Identifying what variables are needed
  - Breaking down the problem into smaller ones
  - Coming up with the idea to solve the problem
  - Coming up with flowcharts (or other forms to present the solution)
  - Check that the solution is correct, and
  - Only finally, write out the code in C

Exercise 1

Question 1: GCD

In Post-Lecture Diagnostic Quiz 2 Problem C on Recursion, the recursive function F computes the greatest common divisor of two positive integers using the Euclidean algorithm (aka Euclid's algorithm).

Complete a program `gcd.c` so that it reads in two positive integers from the standard input and prints out their greatest common divisor.

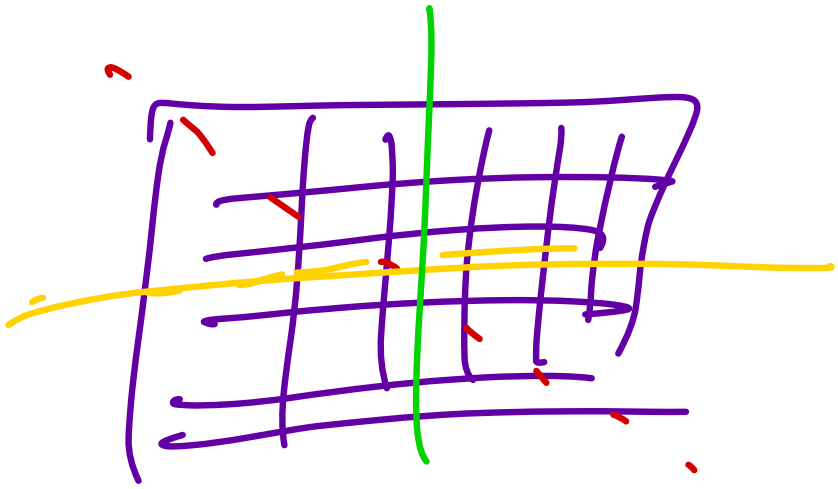
Exercise 1

Question 2: Leap Year

A leap year is a calendar year containing an extra day to synchronize the calendar to seasons and astronomical events. In the Gregorian calendar, years that are multiples of four (except for years divisible by 100 but not by 400) are leap years.

Complete the program `leap.c` so that it reads in an integer representing a year from the standard input and prints out " is a leap year" if the input is a leap year. Otherwise, print " is not a leap year" to the standard output.

Your program should include a `bool` function `is_leap_year` that takes in the input year and returns `true` if the input is a leap year and returns `false` otherwise.



Exercise 1

Question 3: Days Since 1 January

Write a program called `days` that reads in two integers from the standard input, the first is the month (ranged 1 to 12, inclusive) and the second is the day (ranged 1 to 31, inclusive). The program should print to the standard output which day of the year it is. Assume that the year is not a leap year.

Exercise 2

Question 1: Binary

In this question, you are asked to convert a number represented in binary format (using digits 0 and 1) into the decimal format (using digits 0 and 9).

A number in decimal format is represented with base 10. The last digit (rightmost) corresponds to the unit of 1, the next digit (second last) corresponds to the unit of 10, and so on. So, one can write the decimal number, for instance, 7146 as 7×1000 + 1×100 + 4×10 + 6×1.

A number represented in binary uses base 2 instead of base 10. The last digit corresponds to 1 . The second last digit correponds to 2, the third last digit corresponds to 4, and so on. So, the binary number 1101, for instance, corresponds to 1×8 + 1×4 + 1×1 = 13.

Write a program called binary that reads in a positive integer consists of only 0s and 1s from the standard input, treats it as a binary number, and prints the corresponding decimal number to the standard output.

Exercise 2

Question 2: Rectangle

Write a program called rectangle that reads two positive integers from the standard input, corresponding to the width and the height of the rectangle. The width and height must be at least 2. Draw a rectangle on the screen using the special ASCII characters #define "╔" "╗" "╝" "╚" "═" "║", which corresponds to the top left, top right, bottom right, bottom left, top/bottom edge, and left/right edge of the rectangle respectively. Strings consisting of these special characters have been given to you in rectangle.c and we have defined them as constants. For instance, "╔" is called `TOP_LEFT`, and to print this out, you can write

```
cs1010_print_string(TOP_LEFT);
```

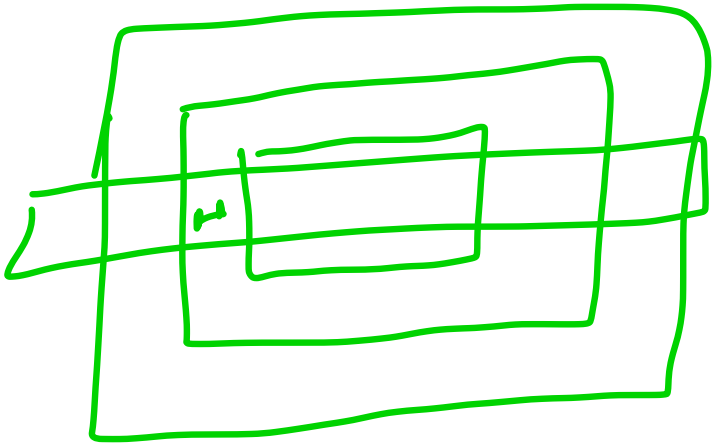
Exercise 2

Question 3: Fibonnaci

The Fibonacci sequence is a sequence of numbers 1, 1, 2, 3, 5, 8, 13, ... Fibonacci numbers often appear in mathematics as well as in nature and have many fascinating properties.

The Fibonacci sequence can be constructed as follows. The first Fibonacci number is 1. The second Fibonacci number is also 1. Subsequently, the i-th Fibonacci number is computed as the sum of the previous two Fibonacci numbers, the (i-2)-th, and the (i-1)-th.

Write a program called fibonacci that reads a positive integer number n from the standard input, and print the n-th Fibonacci number to the standard output. Your program must not use recursion.



Q & A

$x_0 = -3$

$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$ → $|f(x)| = 1$

$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$ $|f(x)| = 0.25$

.

.

.

Q & A

→ $|f(x)| < 0.00000001$

300

x_{300} → root #.