CS1010 Tutorial 3
Group BC1A

10 September 2020

Topics for today

Objectives

- Recap on Topics (Logic, Assertion, Loops)
- Going through problem set 9, 10, 11
- Common issues with Assignment 1
- Assignment 2
- Summary

Items that are graded

- *Assignment 1*
- Assignment 2

Items that are not graded

- *Exercise 0*
- *Exercise 1*
- *Exercise 2*

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)

Requirement

1010

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)

Assignment 2.

- 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*

1 question = 30 mins.

Important details

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

Not Released.

```
if (true) {
    Return 0
} else {
}
```

```
for (long i = 0; ....)
{ Hole sth
    for (long j = 0; ....)
    }
}
```

~~return 1~~

Return 1

Problem 9.1 Question

Given two `bool` variables, `a` and `b`, there are four possible combinations of `true` `false` values. What are the values of `a && b`, `a || b`, and `!a` for each of these combinations? Fill in the table below.

a	b	a && b	a || b	!a
true	true	true.	true	false
true	false	false.	true	false
false	true	false.	true	true
false	false	false.	false.	true.

```
if (!true) {
  // do something.
}
```

a && b = a AND b must be true for output to be true.

short circuiting.

a || b = If either A or B is true, returns true.

!a = Not a (Negate A) → turns true to false false to true.

Problem 9.1 Solution

```
| `a`    | `b`    | `a && b` | `a || b` | `!a`   |
|--------|-------|----------|----------|-------|
| true   | true  | true     | true     | false |
| true   | false | false    | true     | false |
| false  | true  | false    | true     | true  |
| false  | false | false    | false    | true  |
```

Take note of short circuiting

a	b	a && b
true	true	true
true	false	false
false	true	false
false	false	false

a	b	a || b
true	true	true
true	false	true
false	true	true
false	false	false

Problem 9.2 Question

Consider the function below, which aims to return the maximum value given three numbers.

```
long max_of_three(long a, long b, long c)
{
  long max = 0;
  if ((a >= b) && (a >= c)) {
    // a is larger than b and c
    max = a;
  }
  if ((b >= a) && (b >= c)) {
    // b is larger than a and c
    max = b;
  }
  if ((c >= a) && (c >= b)) {
    // c is larger than a and b
    max = c;
  }
  return max;
}
```

A

2 2 → doesn't work

2 1 1 → works.

1 1 2

if ((c==a) || (c==b)) {
// do stuff

(a) What is wrong with the code above?

(b) Give a sample test value of `a`, `b`, and `c` that would expose the bug.

(c) Fix the code above to remove the bug.

(d) Replace the three `if` statements in the code above with `if-else` statements. Draw the corresponding flowchart.

Problem 9.2 Solution

a) This code does not check for equality between numbers, output will be 0

b) a = b = c, max will be 0

A tie among two max number, i.e. 2 2 1, max will also be 0

c)

```
long max_of_three(long a, long b, long c)
{
  long max = 0;
  if ((a >= b) && (a >= c)) {
    // a is larger than b and c
    max = a;
  }
  if ((b >= a) && (b >= c)) {
    // b is larger than a and c
    max = b;
  }
  if ((c >= a) && (c >= b)) {
    // c is larger than a and b
    max = c;
  }
  return max;
}
```

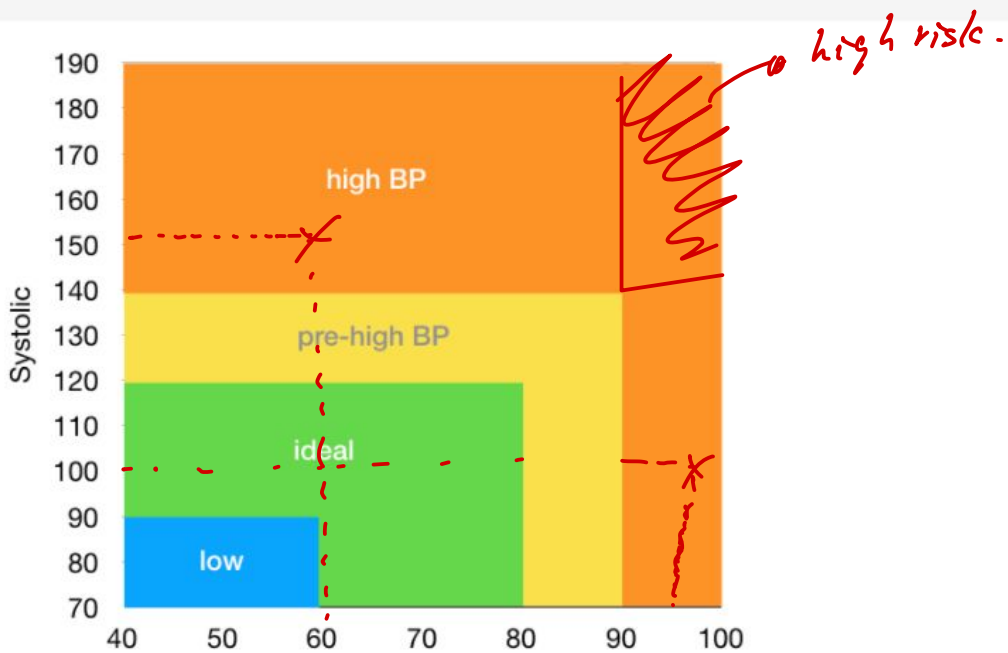
d)

```
long max_of_three(long a, long b, long c)
{
  long max = 0;
  if (a >= b) {
    if (a >= c) {
      max = a;
    } else {
      max = c;
    }
  } else if (b >= c) {
     max = b;
  } else {
    max = c;
  }
  return max;
}
```

Problem 9.3 Question

Write a function that takes in a blood pressure measurement, and prints either `low`, `ideal`, `pre-high`, and `high` depending on the input values. The blood pressure is given as two `long` values, the systolic and the diastolic. The text to be printed depends on the range, depicted in the figure below.

```
void print_blood_pressure(long systolic, long diastolic)
{
  :
}
```



The figure does not say how to classify the data if the values fall exactly on the boundary of two regions. In this case, you can classify it into either region.

Problem 9.3 Answer

- Two approaches

Using || operator

```
void print_blood_pressure(long systolic, long diastolic) {
    if (systolic > 140 || diastolic > 90) {
        cs1010_println_string("high");
    } else if (systolic > 120 || diastolic > 80) {
        cs1010_println_string("pre-high");
    } else if (systolic > 90 || diastolic > 60) {
        cs1010_println_string("ideal");
    } else {
        cs1010_println_string("low");
    }
}
```

Using && operator

```
void print_blood_pressure(long systolic, long diastolic) {
    if (systolic < 90 && diastolic < 60) {
        cs1010_println_string("low");
    } else if (systolic < 120 && diastolic < 80) {
        cs1010_println_string("ideal");
    } else if (systolic < 140 && diastolic < 90) {
        cs1010_println_string("pre-high");
    } else {
        cs1010_println_string("high");
    }
}
```

Topic 10 Assertion

De morgan's law

- `!(e1 && e2)` is the same as `(!e1) || (!e2)`
- `!(e1 || e2)` is the same as `(!e1) && (!e2)`

$\overline{(A \,\&\&\, B)} \;===\; (\bar{A} \,||\, \bar{B})$

$\overline{(A \,||\, B)} \;===\; (\bar{A} \,\&\&\, \bar{B})$

$\overline{(A \,||\, B)}$

A	B	A || B	$\overline{A \,||\, B}$	A && B	$\bar{A}$	$\bar{B}$	$\bar{A}$ && $\bar{B}$
T	F	T	F	F	F	T	F
T	T	T	F	T	F	F	F
F	F	F	T	F	T	T	T
F	T	T	F	F	T	F	F

Problem 10.1 Question

Negate the following logical expression, then apply De Morgan's Law to simplify the resulting expression. Assume all variable names mentioned are boolean variables.

(a) `(x > 1) && (y != 10)`

(b) `!eating && drinking`

(C) `(has_cs2030 || has_cs2113) && has_cs2040c`

a) !((x > 1) && (y != 10)) = !(x > 1) || !(y != 10)

= (x <= 1) || (y == 10)

b) !(!eating && drinking) = eating || !drinking.

c) !((has_cs2030 || has_cs2113) && has_cs2040c)

= !(has_2030 || has_2113) || !has_2040c

= (!has_2030 && !has_2113) || !has_2040c.

Problem 10.1 Answer

a) `!((x > 1) && (y != 10)` => `!(x > 1) || !(y != 10)` => `(x <= 1) || (y == 10)`

b) `!(!eating && drinking)` => `(eating || !drinking)`

c) `!((has_cs2030 || has_cs2113) && has_cs2040c)` =>

`!(has_cs2030 || has_cs2113) || !has_cs2040c` =>

~~`!(has_cs2030 || has_cs2113) || !has_cs2040c`~~ =>

`(!has_cs2030 && !has_cs2113) || !has_cs2040c`

Problem 10.2 Question

In the code below, replace `???` with the appropriate assertion. What will be printed?

if (something == true)

```
long score = 4;
if (something) {
  score = 10;
} else {
  score = 0;
}
// { ??? }

if (score == 4) {
    score = 1;
} else {
    score += 10;
}
// { ??? }

if (score >= 10) {
    cs1010_println_string("ok");
} else {
    cs1010_println_string("failed");
}
```

① - { score == 10 || score == 0 }

② { score == 20 || score == 10 }

→ Always give "ok"

Problem 10.2 Answer

```
long score = 4;
if (something) {
  score = 10;
} else {
  score = 0;
}
// { score == 10 || score == 0 }

if (score == 4) {
    score = 1;
} else {
    score += 10;
}
// score is never 4! So we always execute the false block.
// { score == 20 || score == 10 }

// Now, the score is always >= 10, and "ok" will always be printed.

if (score >= 10) {
    cs1010_println_string("ok");
} else {
    cs1010_println_string("failed");
}
```

Problem 11.1 Question

Here is another version of the `factorial` function:

```
long factorial(long n)
{
    long i = n - 1;
    long product;
    for (product = n; i >= 2; product *= i)
    {
        i -= 1;
    }
    return product;
}
```

4

i = 4 - 1 = 3

pro = ??

pro = 4 true

4 x 3 = 12.

i = 3 - 1 = 2.

2nd loop

product = 12

i = 2

product = 12 x 2 = 24

i = 1

3rd loop

Does this code run correctly? If it is incorrect, explain what is wrong and suggest a fix. (Hint: translate this to the corresponding flowchart and trace through the flowchart).

Problem 11.1 Answer

```
long factorial(long n)
{
    int i = n+1;
    long product;
    for (product = 1; i >= 2; product *= i)
    {
        i -= 1;
    }
    return product;
}
```

Problem 11.2 Question

Guess a number

```c
#include <stdlib.h>
#include <sys/times.h>
#include "cs1010.h"

int main()
{
  // Initialize the random number generator
  srandom(times(0));

  // Generate a random number between 1 and 100
  long answer = (random() % 100) + 1;

  long guess;
  do {
    // Read guess and feedback to user
    guess = cs1010_read_long();
    if (guess > answer) {
      cs1010_println_string("too high");
    } else if (guess < answer) {
      cs1010_println_string("too low");
    }
  } while (guess != answer);

  // { guess == answer }
  cs1010_println_string("you got it.  congrats!");
}
```

Problem 11.2(a) Answer

```
long count = 0;
do {
  // Read guess and feedback to user
  guess = cs1010_read_long();
  count += 1;

  if (guess > answer) {
    cs1010_println_string("too high");
  } else if (guess < answer) {
    cs1010_println_string("too low");
  }
} while (guess != answer);
```

Problem 11.2(b) Answer

```
// Read guess and feedback to user
guess = cs1010_read_long();
while (guess != answer) {
  if (guess > answer) {
    cs1010_println_string("too high");
  } else if (guess < answer) {
    cs1010_println_string("too low");
  }
  // Read guess and feedback to user
  guess = cs1010_read_long();
}
```

-1 20.

do ~~while~~.

while.

Problem 11.2(c) Answer

Make a function

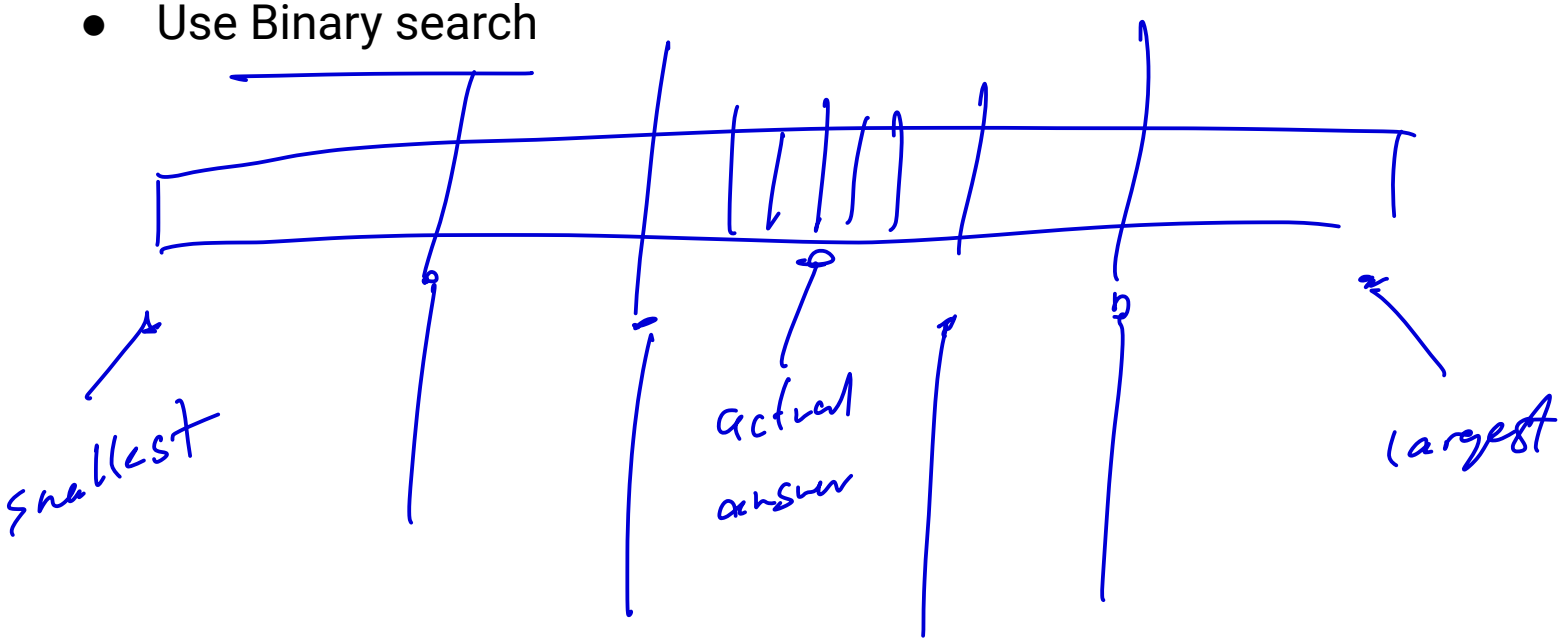
```
long guess_a_number()
{
  // Generate a random number between 1 and 100
  long answer = (random() % 100) + 1;
  long guess;
  long count = 0;
  do {
    // Read guess and feedback to user
    guess = cs1010_read_long();
    count += 1;
    if (guess > answer) {
      cs1010_println_string("too high");
    } else if (guess < answer) {
      cs1010_println_string("too low");
    }
  } while (guess != answer);

  // { guess == answer }
  cs1010_println_string("you got it.  congrats!");
  return count;
}
```

In the outer loop

```
count = 0;
for (i = 0; i < 4; i += 1) {
  count += guess_a_number()
}
cs1010_println_double(count/5.0);
```

Problem 11.2(d) Answer

- Use Binary search



efficiency of $\log_2(n)$

Problem 11.3 Question

Trace the following algorithms:

```
long mystery(long n, long k) {
    long something = n;
    long count = -1;
    while (something >= 1) {
        something /= k;
        count += 1;
    }
    return count;
}
```

8 2

8.

-1

c = 0.

8/2 = 4 = s

s = 4/2 = 2

c = 1

2/2 = 1

c = 2

1/2 = 0

c = 3

(a) What is the return value when

- n is 8 and k is 2?
- n is 81 and k is 3?
- n is 100 and k is 5?

Answer these questions by reading the code first, instead of trying it out on a computer (you can verify later).

(b) What is the mathematical expression that our mystery function here is trying to compute based on the examples above?

n = 0 , -1

(c) Give a pair of inputs that would cause the function to return the wrong answer.

(d) Give a pair of inputs that would cause the function to loop forever.

k = 1

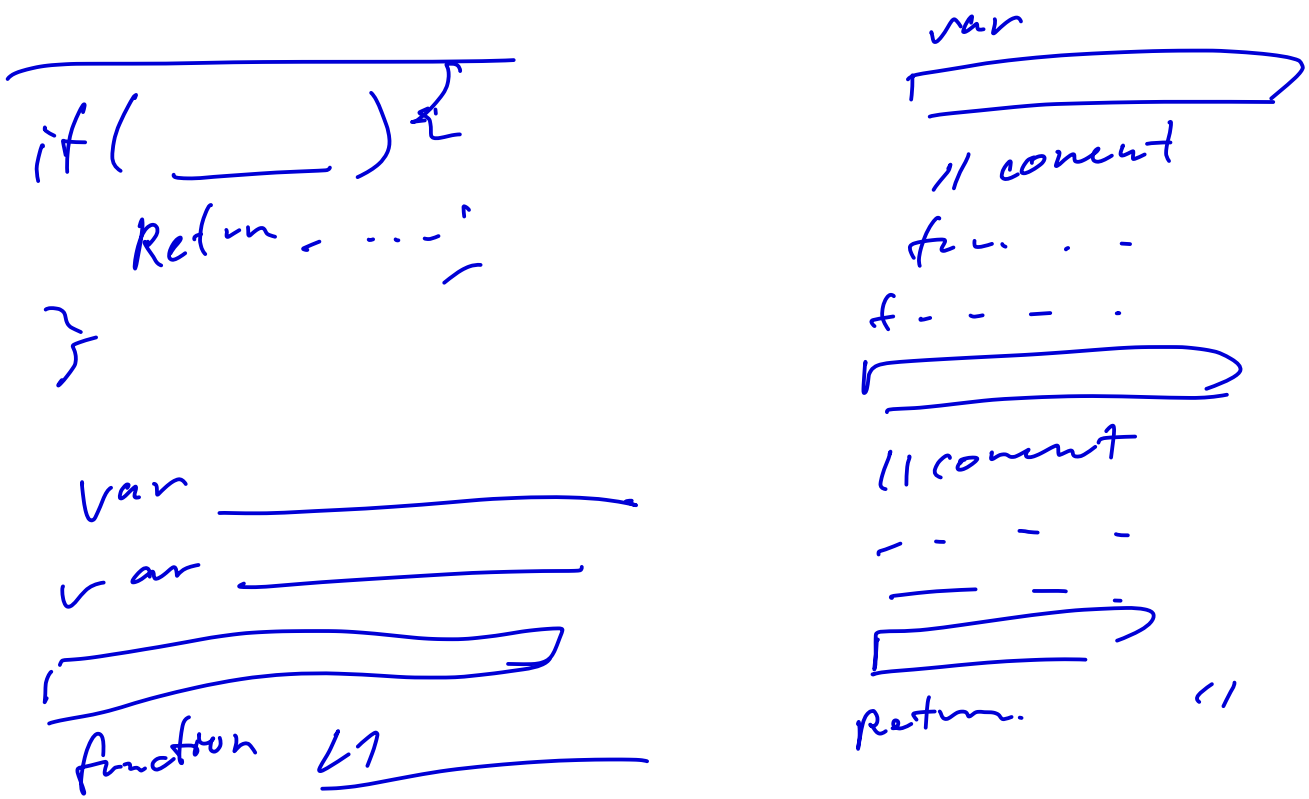
Problem 11.3 Question

$\lfloor \log(kn) \rfloor$

a) 3, 4, 2 Respectively
b) From part a), observe that formula is floor(log(kn))
c) When n = 0, the value returned would be -1
d) When k = 1, the value of n will never change, hence infinite loop

Assignment 1 Issues

- Take note of formatting
  - While formatting is not graded this assignment, it might be graded for the rest of the assignment



function

Assignment 2

- Different format from Assignment 1
  - Questions will all be the file ‘questions.md’
  - use :vsp to open this file side by side to your current working window in terminal
  - use Ctrl + w to move between viewports
  - More commands available here
    - https://www.linux.com/training-tutorials/vim-tips-using-viewports/
- Please use vim as this is the main editor to be used in PE
- Are the color schemes easy to read? If not do feedback to me

comments not readable.
vs dark blue.
dynamic color scheme?