

# CS1010

[https://github.com/DigiPie/cs1010\\_tut\\_c09](https://github.com/DigiPie/cs1010_tut_c09)

# Today's plan

- Tutorial Segment
  - *Unix & Clang*
  - *Recap on C*
  - *Discussion of problem sets*
- Lab Segment
  - *Programming Assignment (PA0)*

The background of the image is a stylized world map divided into four quadrants by a vertical and a horizontal line. The top-left quadrant is red, the top-right is blue, the bottom-left is yellow, and the bottom-right is green. The word "Kahoot!" is written in a large, white, sans-serif font across the center of the image, with the exclamation mark positioned at the end of the word in the green quadrant.

Kahoot!

# Unix commands

- <https://nus-cs1010.github.io/1819-s1/unix/index.html>
- <http://cheatsheetworld.com/programming/unix-linux-cheat-sheet/>

# Compiling with Clang

clang -Wall -g teh.c -o teh -lm

- What is -Wall? *Enable Warnings All*
- What is -g? *Generate additional info for lldb debugger*
- What is -o teh? *Output file name set to teh*
- What is -lm? *Link to Math library*

<https://nus-cs1010.github.io/1819-s1/clang/index.html>

# Today's plan

- Tutorial Segment
  - *Unix & Clang*
  - *Recap on C*
  - *Discussion of problem sets*
- Lab Segment
  - *Programming Assignment (PA0)*

# Recap on C

CS1010 Tut [C09]

```
1  #include <math.h>
2  #include "cs1010.h"
3
4  long square(long x)
5  {
6      return x*x;
7  }
8
9  double hypotenuse_of(long base, long height)
10 {
11     return sqrt(square(base) + square(height));
12 }
13
14 int main()
15 {
16     double hypotenuse;
17     long base = cs1010_read_long();
18     long height = cs1010_read_long();
19     hypotenuse = hypotenuse_of(base, height);
20     cs1010_println_double(hypotenuse);
21 }
```

# Today's plan


- Tutorial Segment
  - *Unix & Clang*
  - *Recap on C*
  - *Discussion of problem sets*
- Lab Segment
  - *Programming Assignment (PA0)*





# PROBLEM SETS

3.1, 3.2, 5.1, 5.2



# Problem Set 3.1

$$\frac{\sum_{i=0}^{k-1} |l_i - \mu|}{k}$$

- MAD: How spread out a set of data is.
- Absolute deviation: Absolute difference between an element and the mean.
- MAD: Mean of all absolute deviations.

# Problem Set 3.1

$$\frac{\sum_{i=0}^{k-1} |l_i - \mu|}{k}$$

- Given the following functions: `mean(L, k)`, `subtract(L, k, s)`, `abs(L, k)`, how do you solve this problem?
  - *L*: List of elements; *k*: Number of elements in *L*
  - *mean(L, k)*: Returns mean of list *L*
  - *subtract(L, k, s)*: Subtracts *s* from every element in *L*
  - *abs(L, k)*: Absolutes every element in *L*

# Problem Set 3.1

$$\frac{\sum_{i=0}^{k-1} |l_i - \mu|}{k}$$

- Find the mean:
  - *mean(L, k)*
- Subtract the mean from every element
  - *subtract(L, k, mean(L, k))*

# Problem Set 3.1


$$\frac{\sum_{i=0}^{k-1} |l_i - \mu|}{k}$$

- Absolute every element in the list:
  - `abs(subtract(L, k, mean(L, k)), k)`
- Find the mean again
  - `mean(abs(subtract(L, k, mean(L, k)), k), k)`



# PROBLEM SETS

3.1, 3.2, 5.1, 5.2



# Problem Set 3.2 (a)

Find the sum of all integers in L recursively

```
sum(L, i, j) {  
    if ( i == j ) {  
        return Li;  
    } else {  
        return Li + sum(L, i+1, j);  
    }  
}
```

## Problem Set 3.2 (b)

The function `pow(i,j)` computes  $i^j$ . How do we compute recursively?

- `pow(2,3)` returns  $2^3 = 2 \times 2 \times 2 = 8$



```
#include "cs1010.h"
```

```
int recursive_pow(int integer, int exponent)
{
    if (exponent == 1)
    {
        return integer;
    }
    else
    {
        return integer * recursive_pow(integer, --exponent);
    }
}
```


```
int main()
{
    cs1010_print_string("recursive_pow(2, 3) aka 2^3: ");
    int output = recursive_pow(2, 3);
    cs1010_println_long(output);
}
```

```
~
~
~
```



# PROBLEM SETS

3.1, 3.2(b), 5.1, 5.2



# Problem Set 5.1

Would passing an int into `sqrt(...)` result in an error?

```
double sqrt(double x);
```

```
double  
hypotenuse_of(long base, long height)  
{  
    return sqrt(square(base) + square(height));  
}
```

# Problem Set 5.1

No, because any double can hold any int.

```
double sqrt(double x);
```

```
double  
hypotenuse_of(long base, long height)  
{  
    return sqrt(square(base) + square(height));  
}
```

~~~~~



sunfire-r.comp.nus.edu.sg - PuTTY



```
evantay@pe118:~/code/tut2$ ./problem5_1
Double can hold any integer value without data loss
Printing int i = 23 as double: 23.0000
Converting double to integer results in data loss
Printing double d = 3.142 as integer: 3
evantay@pe118:~/code/tut2$
```



# CODE AVAILABLE HERE


[https://github.com/DigiPie/cs1010\\_tut\\_c09](https://github.com/DigiPie/cs1010_tut_c09)





# PROBLEM SETS

3.1, 3.2(b), 5.1, 5.2





# Problem Set 5.2

What return type should we use?

```
1  ??? square(uint16_t x) {  
2      return x*x;  
3  }
```

- Max value of input is  $2^{16} - 1 = 65,535$ 
  - *(limited by **uint16\_t** parameter)*

# Problem Set 5.2

What return type should we use? **Must be big enough to store all possible values of  $x * x$**

```
1  ??? square(uint16_t x) {  
2      return x*x;  
3  }
```

- Max value of input is  $2^{16} - 1 = 65,535$ 
  - *(limited by **uint16\_t** parameter)*
- Can we use `uint16_t` for return type?

# Problem Set 5.2

What return type should we use? **Must be big enough to store all possible values of  $x * x$**

```
1  ??? square(uint16_t x) {  
2      return x*x;  
3  }
```

- Max value of input is  $2^{16} - 1 = 65,535$ 
  - *(limited by **uint16\_t** parameter)*
- Can we use uint16\_t for return type? **No**
  - *What if input was 65, 535?*

# Problem Set 5.2

What can the return type of this square be, in order for the return type to be big enough to store all possible values of  $x * x$ ?

```
1  ??? square(uint16_t x) {  
2      return x*x;  
3  }
```

- Max value of input is  $2^{16} - 1 = 65,535$ 
  - (limited by ***uint16\_t*** parameter)
- Can we use ***uint32\_t*** for return type?

# Problem Set 5.2

What can the return type of this square be, in order for the return type to be big enough to store all possible values of  $x * x$ ?

```
1  ??? square(uint16_t x) {  
2      return x*x;  
3  }
```

- Max value of input is  $2^{16} - 1 = 65,535$ 
  - (limited by **uint16\_t** parameter)
- Can we use uint32\_t for return type? **Yes**
  - $(2^{32}) - 1 > ((2^{16}) - 1)^2$

# Problem Set 5.2

What can the return type of this square be, in order for the return type to be big enough to store all possible values of  $x * x$ ?

```
1  ??? square(uint16_t x) {  
2      return x*x;  
3  }
```

- Max value of input is  $2^{16} - 1 = 65,535$ 
  - *(limited by **uint16\_t** parameter)*
- Can we use `int32_t` for return type?

# Problem Set 5.2

What can the return type of this square be, in order for the return type to be big enough to store all possible values of  $x * x$ ?

```
1  ??? square(uint16_t x) {  
2      return x*x;  
3  }
```

- Max value of input is  $2^{16} - 1 = 65,535$ 
  - (limited by **uint16\_t** parameter)
- Can we use int32\_t for return type? **No**
  - Range is from  $-(2^{16})$  to  $(2^{16}) - 1$

# Today's plan

- Tutorial Segment
  - *Unix & Clang*
  - *Recap on C*
  - *Discussion of problem sets*
- Lab Segment
  - ***Exercise 1: Freezer.c***





Q & A



# THE END

[https://github.com/DigiPie/cs1010\\_tut\\_c09](https://github.com/DigiPie/cs1010_tut_c09)

