

# 01204211 Discrete Mathematics

## Lecture 1: Introduction

Jittat Fakcharoenphol

August 16, 2015

# The goals of this course

There are two goals:

- ▶ To learn how to make mathematical arguments.

# The goals of this course

There are two goals:

- ▶ To learn how to make mathematical arguments.
- ▶ To learn various fundamental mathematical concepts that are very useful in computer science.

# What is mathematics?

# What is mathematics?

Ah... that's a philosophical question.

# What is mathematics?

Ah... that's a philosophical question.

IMHO, mathematics is a mean to communicate *precise* ideas.

It's like learning a new language

# Why should we learn how to prove? (1)



# Why should we learn how to prove? (1)

Look at this program.

```
if a > b:  
    return a  
else:  
    return b
```

The author claims that this program takes two variables  $a$  and  $b$  and returns the larger one.

# Why should we learn how to prove? (1)

Look at this program.

```
if a > b:  
    return a  
else:  
    return b
```

The author claims that this program takes two variables  $a$  and  $b$  and returns the larger one.

*Do you believe the author of the code?*

# Why should we learn how to prove? (1)

Look at this program.

```
if a > b:  
    return a  
else:  
    return b
```

The author claims that this program takes two variables  $a$  and  $b$  and returns the larger one.

*Do you believe the author of the code? Why?*

## Why should we learn how to prove? (2)

Now look at this program.

```
if a > b:
    if a > c:
        return a
    else:
        return c
else:
    if c > b:
        return c
    else:
        return b
```

The author claims that this program takes three variables  $a$ ,  $b$  and  $c$  and returns the largest one.

## Why should we learn how to prove? (2)

Now look at this program.

```
if a > b:
    if a > c:
        return a
    else:
        return c
else:
    if c > b:
        return c
    else:
        return b
```

The author claims that this program takes three variables  $a$ ,  $b$  and  $c$  and returns the largest one.

*Do you believe the author of the code?*

## Why should we learn how to prove? (2)

Now look at this program.

```
if a > b:
    if a > c:
        return a
    else:
        return c
else:
    if c > b:
        return c
    else:
        return b
```

The author claims that this program takes three variables  $a$ ,  $b$  and  $c$  and returns the largest one.

*Do you believe the author of the code? Why?*

## Why should we learn how to prove? (3)

Finally, look at this program.

```
// Input: array A with n elements: A[0], ..., A[n-1]
m = A[0]
for i = 1, 2, ..., n-1:
    if A[i] > m:
        m = A[i]
return m
```

The author claims that this program takes an array  $A$  with  $n$  elements and returns the maximum element.

## Why should we learn how to prove? (3)

Finally, look at this program.

```
// Input: array A with n elements: A[0], ..., A[n-1]
m = A[0]
for i = 1, 2, ..., n-1:
    if A[i] > m:
        m = A[i]
return m
```

The author claims that this program takes an array  $A$  with  $n$  elements and returns the maximum element.

*Do you believe the author of the code?*



## Why should we learn how to prove? (3)

Finally, look at this program.

```
// Input: array A with n elements: A[0], ..., A[n-1]
m = A[0]
for i = 1, 2, ..., n-1:
    if A[i] > m:
        m = A[i]
return m
```

The author claims that this program takes an array  $A$  with  $n$  elements and returns the maximum element.

*Do you believe the author of the code? Why?*

## Why should we learn how to prove? (3)

Finally, look at this program.

```
// Input: array A with n elements: A[0], ..., A[n-1]
m = A[0]
for i = 1, 2, ..., n-1:
    if A[i] > m:
        m = A[i]
return m
```

The author claims that this program takes an array  $A$  with  $n$  elements and returns the maximum element.

*Do you believe the author of the code? Why?*

**Can we try to test the code with all possible inputs?**