

# C/CPS 506

Assignment Project Exam Help

Compara <https://eduassistpro.github.io/> Languages

Prof. Ale  
Add WeChat edu\_assist\_pro

**Topic 7:** Pattern matching, custom types in Haskell

# Notice!

---

**Obligatory copyright notice in the age of digital  
delivery and online classrooms:  
Assignment Project Exam Help**

*The copyright to this work belongs to Alex Ufkes. Students registered in course COP5506 can use it for the purposes of this course but no other use is permitted, and there can be no sale or transfer or use of the work for any other purpose without explicit permission of Alex Ufkes.*

# Course Administration

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro  
It's important to start                    bout the  
assignment if you haven't already.

# Any Questions?

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Let's Get Started!

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Types in Haskell

---

## Statically Typed:

- Haskell uses static type checking.
- Every expression is assigned a type.
- If a function expected type, a com <https://eduassistpro.github.io/>

## Type Inference Add WeChat edu\_assist\_pro

- Like Python, and unlike Java, we need not specify type.
- It is inferred by the context: X = “Hello”, X is a string.
- However, we can explicitly specify types.
- Good practice when we know what types we want; compiler will give errors upon type mismatch.

# Types in Haskell

---

:t can be used to reveal type:

Assignment Project Exam Help

<https://eduassistpro.github.io/>  
of Num type class.  
1.0 of Fractional type class.

Add WeChat edu\_assist\_pro



- ‘a’ is a **Char**
- “Hello” is a **[Char]**
- **[Char] = String**
- t is a **Bool**

# Num p => p ?

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

p is a type variable  
Add WeChat edu\_assist\_pro

e value can be any  
type in type class (Integer, Float, etc.)

- Haskell is free to treat 7 as it sees fit, so long as it does so in a way that adheres to the operations defined in type class Num.

# Typeclasses?

---

Assignment Project Exam Help

<https://eduassistpro.github.io/types/generic-as-possible>

- If we explicitly type a variable as integer, it can't be promoted to a float.
- However, if we generically infer it to be a Num, it can be used anywhere any other member of Num is allowed.

# Types in Haskell

---

We can explicitly indicate types:

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

```
Prelude> :t 5::Int
5::Int :: Int
Prelude> :t 5.0::Double
5.0::Double :: Double
Prelude> |
```

- Us                    a type
- My advice for you is to start by letting the inference engine figure it out.
- **At this point, it knows better than you.**

# Types in Haskell

---

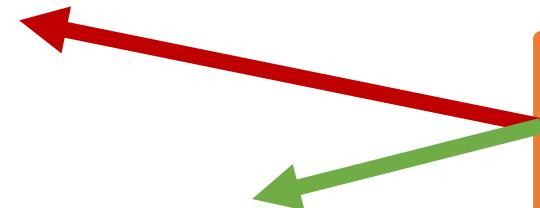
We can explicitly indicate types:

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

When defining a name, can indicate type explicitly:



# Type Classes

---

Type polymorphism and type variables:

## ~~Recall Assignment~~ Project Exam Help

- In language  $\alpha$ , a function  $f$  is overloaded if it has different implementations for different types.
- Numeric type equality and inequality operations are performed differently.
- In general, if we want to compare two values of type  $\alpha$ , we use an  $\alpha$ -compare
- $\alpha$  is a *type variable*, because its value is a type.

# Type Classes

---

Consider the equality (==) operator:

Takes two parameters, each of the same type (call it  $\alpha$ ), and returns a Boolean

This operator <https://eduassistpro.github.io/> pes, just some.

Add WeChat edu\_assist\_pro

Thus, we can associate == with a specific *type class* containing those types for which == is defined.

This type class is called **Eq** in Haskell.

# Eq Type Class

---

(==) is defined for types

in typeclass Eq

Assignment Project Exam Help

<https://eduassistpro.github.io/>

- $(==)$  takes two arguments of type **a**, where **a** is a type class **Eq**
- It returns **Bool**

- If a concrete type, **a**, belongs to a certain type class, we say **a** is an *instance* of that type class.
- **Int** is an instance of **Eq**, for example.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Num Type Class

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat `edu_assist_pro`

- This allows numbers freedom to be an integer or floating point as the compiler sees fit.
- **Num** class contains all numbers, and certain operations over them such as addition.

# Num Type Class

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat ~~edu\_assist\_pro~~

- **p** is a type class
- The type of 5 is **p**, and **p** is a member of type class **Num**

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Show Type Class

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat  
T<sub>re</sub> members of the  
**Show** class have functions which  
convert their value to a String.

# Function Types

---

Assignment Project Exam Help

↳ <https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro  
tai containing type a,  
and returns a list containing type a

a and b can be *literally any type!*

# Function Types

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat `edu_assist_pro`

- Function returns `IO action`.
- `ut, IO()` is output.
- More on IO actions later.

Same as [Char]

We'll create our own types soon, and see how to  
Assignment Project Exam Help  
add t lasses.

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Functions in Haskell

---

As expected of a pure functional language, functions are central in Haskell

Assignment Project Exam Help

<https://eduassistpro.github.io/>

If we're compiling our  
Add WeChat `edu_assist_pro` de into an executable,  
e need a main.

- If we're using the GHCi shell, we don't.

# Functions in Haskell

---

Let's start simple:

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Define function called **square**  
that takes one argument **x**

Add WeChat **edu\_assist\_pro**, where  
**x** is the input argument

# Functions in Haskell

---

Let's start simple:

Assignment Project Exam Help

<https://eduassistpro.github.io/> function square  
Add WeChat edu\_assist\_pro typical fashion

# Functions in Haskell

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro



# Functions in Haskell

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

19

Prelude>

Passing in four args

# Functions in Haskell

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat [edu\\_assist\\_pro](https://edu_assist_pro)...  
hat we're doing and adding)...

- Haskell determined that input and output type should be instances of typeclass **Num**.
- `(+)` and `(*)` are both defined for all types in typeclass **Num**.

# Haskell Modules

---

This is getting tedious to type interactively.

## Assignment Project Exam Help

<https://eduassistpro.github.io/>

- Let's create a module!
- Add WeChat.edu\_assist\_pro
- Similar to Elixir
  - We can load the module in GHCi
  - Access its functions and expressions

# Loading a Module

---



Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

Assignment Project Exam Help



<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro





When we make changes to Test module, can reload with 1 click!

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Loading a Module

---

Use `:load` in terminal GHCi:

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro



# Control Structures

---

if then else

Assignment Project Exam Help

```
f 67  
1  
Prelude> f 0  
1  
Prelude> f (-6)  
0  
Prelude> f -6
```

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

- Brackets required around negative arguments
- Otherwise it thinks you're subtracting 6 from f

```
<interactive>:203:1: error:  
  • Non type-variable argument in the constraint  
    a:Int, Num (a -> p)
```

# Control Structures

---

**if then else if then else**

Assignment Project Exam Help

- Here we have a function named takes one argument x

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

- 0 if x is 0

- If/else construct in Haskell is similar to most other languages.
- It must include a **then** and an **else**

# Control Structures

---

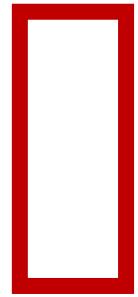
**if then else if then else**

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

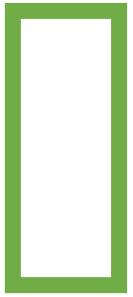
We can now format across multiple lines.  
**HOWEVER:** Indentation matters in Haskell!



Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro



Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro



# Indenting in Haskell

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>  
part of some expression  
Add WeChat edu\_assist\_pro  
sho  
begi  
ted further than the  
t expression

<https://en.wikibooks.org/wiki/Haskell/Indentation>

If all that weren't enough, Tabs don't work  
properly u <https://eduassistpro.github.io/>  
[Assignment](#) [Project](#) [Exam](#) [Help](#)  
aces exactly.

Add WeChat edu\_assist\_pro

# Local Names in Functions

---

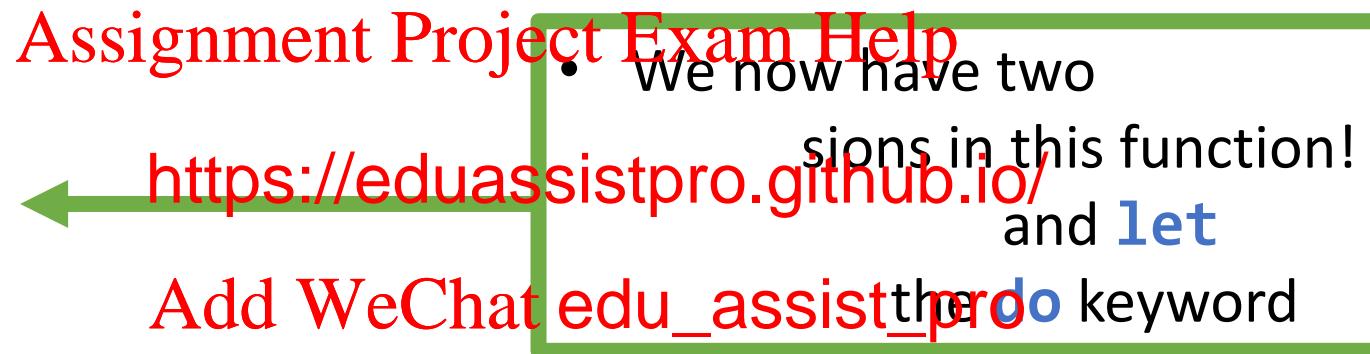
Assignment Project Exam Help



<https://eduassistpro.github.io/>  
ame inside a function  
Add WeChat edu\_assist\_pro or mod the “let” keyword

# Multiple Expressions

---



Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro



# Case Statement

---

Assignment Project Exam Help

<https://eduassistpro.github.io/values>, a  
case Just like easier to write.  
Add WeChat edu\_assist\_pro \_ is wild. It catches  
everything else.

# Case Statement

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Pattern Matching: Case

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

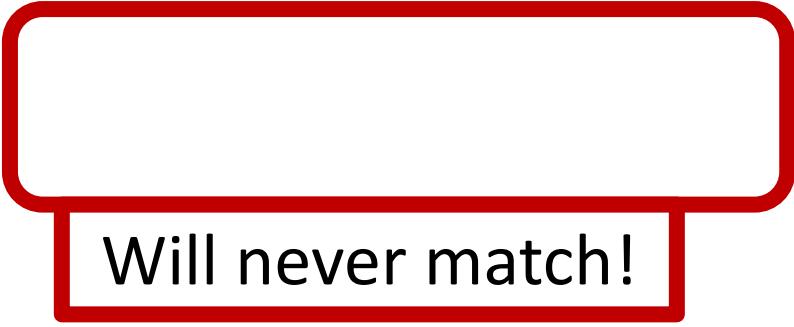
# Pattern Matching: Case

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

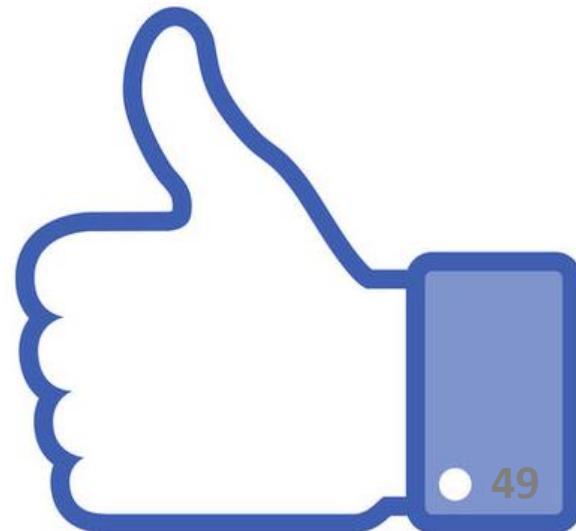


Will never match!

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro



# Unlike Elixir...

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro



Try and be more general to catch anything that isn't a 3-tuple?

# Piecewise Functions

---

Just like Elixir's function signature pattern matching

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Piecewise Functions

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro



- Return unit vector if point lies on axis
- Return input point otherwise

# Functions: Guards

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

- Matches input arguments to **x** and **y**
- Guards denoted with **|**
- **otherwise** is the same as saying **True**

# Recursion

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

Classic list  
length finder

Built-in length function

Our own user function

# Tail Recursion?

---

## Less Assignment Project Exam Help

- In Haskell, tail recursion is different.
- Functions can create a new stack frame.
- In practice, tail recursion is a good deal.

# Recursion: cons

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

Here we treat the input argument as a pair containing the head and tail of the list.

# Recursion: filter

---

- Returns true if  $x \geq 0$
- False otherwise

Assignment Project Exam Help

<https://eduassistpro.github.io> is a Boolean function

- Seco list
- Base case: if the list is empty
- Otherwise, we call the function p with the head of the list.
- If true, append it to the running list
- If false, do not append
- In both cases, make the recursive call with the tail.

# Recursion: filter

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

Test pos function

# Return Multiple *Things*?

---

Lists/tuples to the rescue!

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat

there is a lot of  
computation here.  
• Add a local variable?

# let/in Structure

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro  
• Add do keyword

e expres  
sed to add do keyword

let/in)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Infix Functions

---

Use symbolic operators as functions:

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro



Enclose in parentheses to  
use in non-infix mode

# Function Composition

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

Can be written as:

In math,  $f \circ g$  means “*f following g*”. Same thing in Haskell.

# Lambda Functions

---

Like anonymous functions in Elixir:

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat `edu_assist_pro`  
function with two args

# Lambda Functions

---

They don't need names!

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

True no matter what

Return opposite of input

Good for passing as arguments when  
that's the only place you need them

*Good for passing as arguments when that's the only place you need them*

## Assignment Project Exam Help

<https://eduassistpro.github.io/>  
n of list elements

Add WeChat edu\_assist\_pro

# Comments

---

About time we mentioned them...

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Type Inference

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

Remember this error?



Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro



[Assignment](#) [Project](#) [Exam](#) [Help](#)

**Type of chkClr is inferred:**

- Takes as input a tuple with three values

<https://eduassistpro.github.io/> value in the tuple must

Add `WeChat` to `edu_assist_pro` be inst e class `Eq` and `Num`.  
• The ou lr is a character list.

# Type Inference

---

Assignment Project Exam Help  
Based on the contents of chkClr:  
Haskell determined that the output is [Char]

tuple whose elements must  
<https://eduassistpro.github.io/>  
Num and Eq.

Add WeChat edu\_assist\_pro

```
*Test> :t chkClr
chkClr
  :: (Eq a1, Eq a2, Eq a3, Num a1, Num a2, Num a3) =>
     (a1, a2, a3) -> [Char]
*Test> |
```

# Specify Function Type

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

- **chkAxis** takes a pair-tuple of Floats as input, and returns the same as output.
- Instead of constants being of type Num or Fractional, they are treated as Floats

# Specify Function Type

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Thoughts?

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro



Assignment Project Exam Help  
\*Test>

<https://eduassistpro.github.io/>

### Ord is a type class: Add WeChat edu\_assist\_pro

- When we didn't explicitly define our types, Haskell inferred the type for us.
- Ord is a type class under which the operations used on our inputs are defined.
- I.e., comparison operators.

# Type VS Type Class

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

- Int & Char are types, **not** type classes
- We can use the above notation

- Ord is a type class, thus we specify that **a** is an instance of Ord
- **cmp2** accepts two instances of Ord as arguments.
- Ord contains many different types, **a** can be any of them

# Ord Type Class

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro



- Ord is a type class that supports comparison
- Comparison is all we're doing in our function
- Thus, Haskell infers types as Ord

# Ord Type Class

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

Int is an instance of Ord type class, so when we made our function args explicitly Int, we were OK

# How About This?

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

Assignment Project Exam Help

<https://eduassistpro.github.io/>  
class does not  
parison!

Add WeChat edu\_assist\_pro

# Hmmmm...

---

**Num** doesn't have comparison, **Ord** doesn't have addition

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

It compiled and loaded, what type did Haskell infer for x and y?

## Assignment Project Exam Help

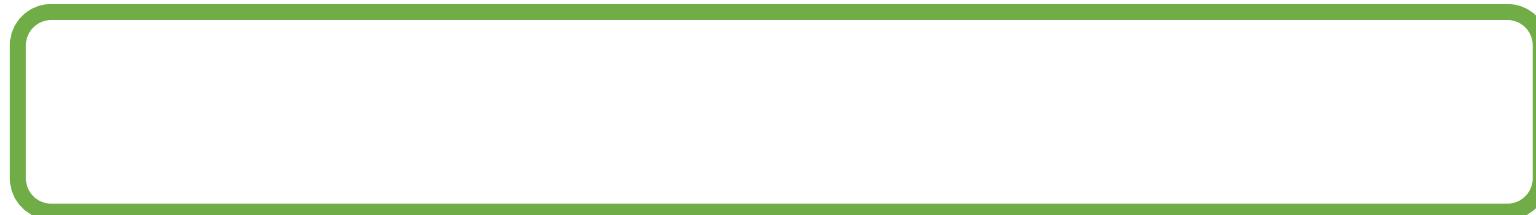
<https://eduassistpro.github.io/>

Both!

Add WeChat edu\_assist\_pro

- Whatever type we pass in (a), it must be an instance of both Ord and Num.
- **Int** is one such type, as is **Float**

**Ord:**



~~Assignment Project Exam Help~~

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

**Num:**



# Custom Data Types

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Custom Data Types

---

- Lists and tuples are already quite powerful for organizing data
- What if we want to add custom behavior over our data?  
Assignment Project Exam Help
- For example, we can define addition for tuples (1, 2).
- What if we want https://eduassistpro.github.io/ to work with tuples and compute the sum? The dot product? Etc.?
- Addition is not defined for tuple Add WeChat edu\_assist\_pro more complicated operations.

# Custom Coordinate Types

---

```
data Pt3 = Assignment Project Exam Help  
          Float Float
```

https://eduassistpro.github.io/  
Add WeChat edu\_assist\_pro

Keyword indicating a custom type definition

Custom type name

Constructor for our custom type.  
To construct a Pt3, we need 3 values of type Float

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Custom Type Usage

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

Echoing a value in the interactive window  
requires its type to be an instance of Show!

# Hmm...

---

- The values contained in Pt3 are Float, and we know that Float is an instance of Show.
- How can we access the individual elements of Pt3?

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

## Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

- Three access functions, one for each of the three values.
- Take as arguments Pt3 (and by extension its three members)
- Return x, y, or z coordinate respectively.

# Overloading Constructor

---

Assignment Project Exam Help

- Define Pt3 with three parameters

<https://eduassistpro.github.io/> with two/parameters  
r data type is

Add WeChat<sup>b</sup> [edu\\_assist\\_pro](https://edu_assist_pro) made it more generic.

There is now a problem with our access functions

*There is now a problem with our access functions.*

## Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

Now our access  
functions work for  
both Pt2 and Pt3



# Deriving Show

---

**Recall:**

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Deriving Show

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

Our custom type will inherit some default display behavior from **Show**

Similar to the `toString()` method in Java!

# More Advanced Functions

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

Compute length of Pt2 and Pt3,  
treating them as vectors

# Addition, Subtraction, Equality?

---

Assignment Project Exam Help

[https://eduassistpro.github.io/  
functions!](https://eduassistpro.github.io/functions!)

- Add WeChat [edu\\_assist\\_pro](https://edu_assist_pro)
- We can easily define addition as the sum of each respective X and Y coordinate.
  - Likewise for subtraction and equality.

# Addition, Subtraction, Equality?

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Addition, Subtraction, Equality?

---

This seems very clunky. Why can't we simply add, subtract, or check equality with the symbolic operators (+, -, ==)?

We can! Equal <https://eduassistpro.github.io/>  
+,-, etc. are defined for instances of type class Eq  
Add WeChat [edu\\_assist\\_pro](https://edu_assist_pro).

How do we make Pt2 and Pt3 instances of another type class?

# Custom Types & Type Classes

---

Assignment Project Exam Help

○ <https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

ine what it means for two Pt2  
lues to be considered equal

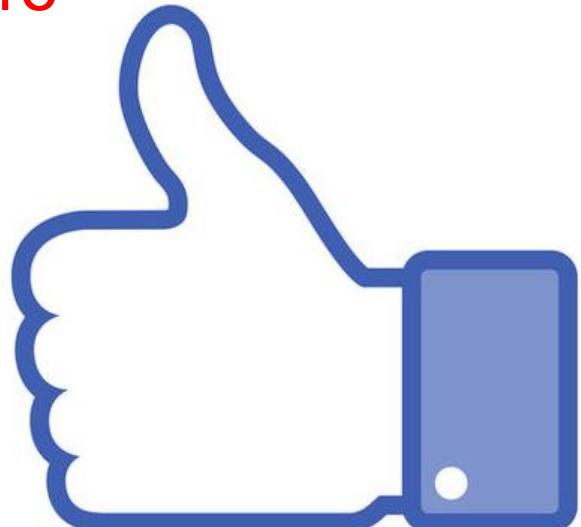
Declare **Pt** to be  
an instance of **Eq**



Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro



# Minimal Definition

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

0 Add WeChat edu\_assist\_pro

- The minimal definition for being an instance of **Eq** is == \*OR\* /= (not equal)
- We only defined ==

## Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

Haskell is clever enough to derive `/=` from our definition of `==`, and vice versa.

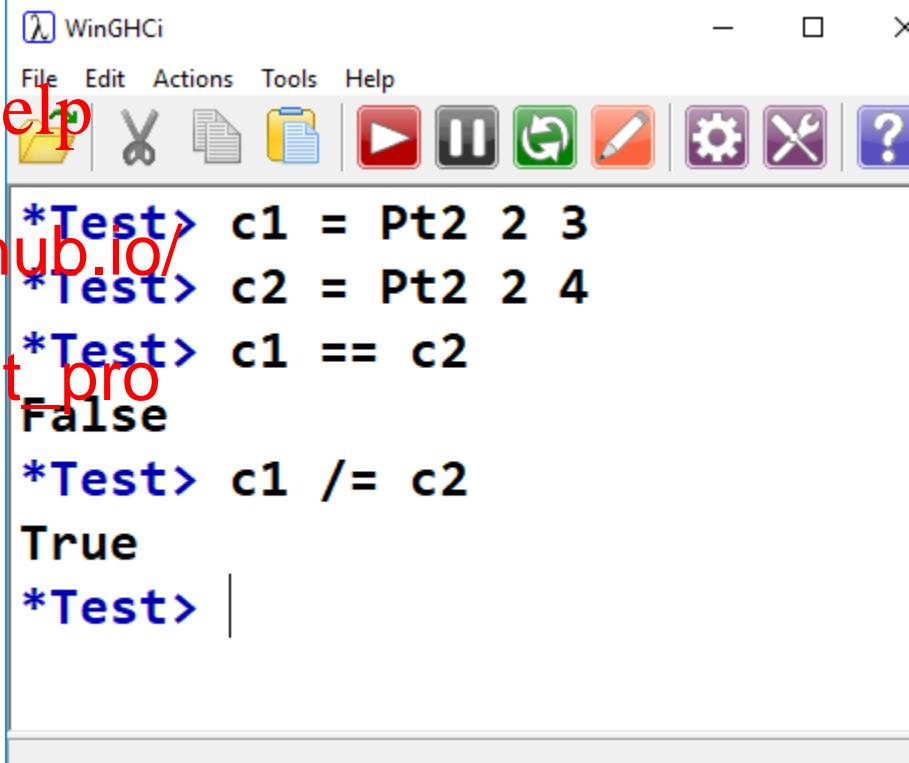
# Let's Add /= Anyway

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

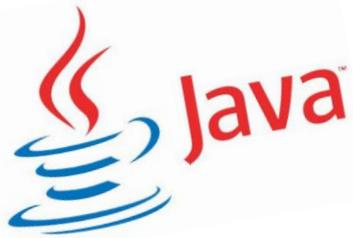


The screenshot shows a Windows application window titled "WinGHCi". The menu bar includes "File", "Edit", "Actions", "Tools", and "Help". The toolbar contains various icons for file operations like cut, copy, paste, and save, along with other tools. The main GHCi prompt area displays the following session:

```
*Test> c1 = Pt2 2 3
*Test> c2 = Pt2 2 4
*Test> c1 == c2
False
*Test> c1 /= c2
True
*Test>
```

# By The Way...

---



This should remind you of interfaces in Java  
**Assignment Project Exam Help**

If you create a Java interface, it must define all methods  
<https://eduassistpro.github.io/> that interface

Add WeChat `edu_assist_pro`

Most common is the **Comparable** interface

If our Java class implements Comparable, we must define some way of comparing two instances of our class

# Instance of Num

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

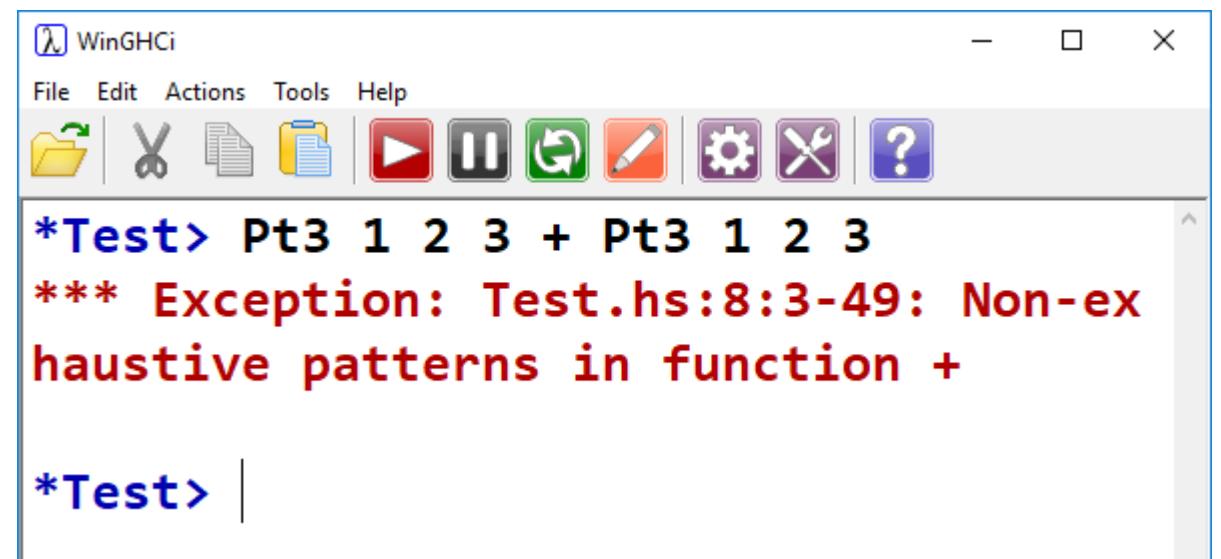
Add WeChat edu\_assist\_pro

## Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

- We're only implementing for Pt2.
- Adding Pt3 follows the same pattern



The screenshot shows a Windows application window titled "WinGHCi". The menu bar includes "File", "Edit", "Actions", "Tools", and "Help". The toolbar contains icons for file operations like Open, Save, Copy, Paste, and a play/pause button. The GHCi prompt is at the top: "\*Test>". Below it, the user has typed "Pt3 1 2 3 + Pt3 1 2 3" and received an error message: "\*\*\* Exception: Test.hs:8:3-49: Non-exhaustive patterns in function +". The cursor is at the bottom prompt, ready for the user's next input.

# Instance of Num

---

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

## Assignment Project Exam Help

<https://eduassistpro.github.io/> d signum/are defined for Float  
fining them for Pt2

Add WeChat edu\_assist\_pro

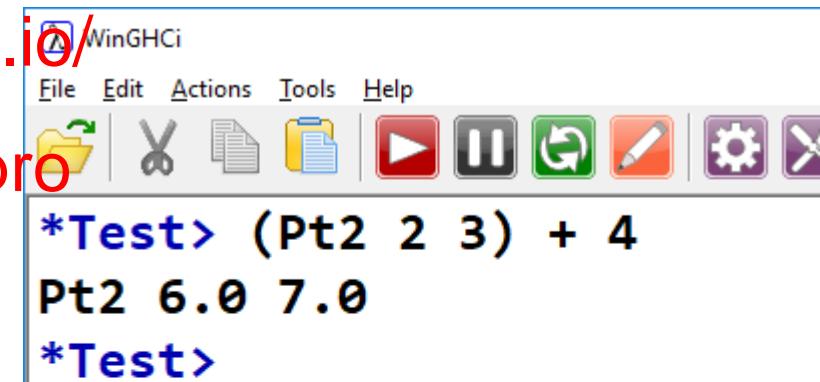
- This may look circular
- We're using abs and signum in our definition of abs and signum.
- er! x1 and y1 are Float.

- `fromInteger` is a *coercion* function.
- Dictates how our custom type can be created from an Integer
- Takes an Integer, returns a Pt
- Lets us do this...

Assignment ~~Project Exam Help~~

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro



The screenshot shows the WinGHCI interface. The menu bar includes File, Edit, Actions, Tools, and Help. The toolbar contains icons for file operations like Open, Save, Cut, Copy, Paste, and others. The GHCi prompt is at the top, followed by the text "WinGHCI". The session window displays the following code and output:

```
*Test> (Pt2 2 3) + 4
Pt2 6.0 7.0
*Test>
```

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

No more  
warnings!

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Instance of **Show**

---

In Java-speak, define our own `toString()`, instead of deriving the default

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat `edu_assist_pro`

- al definition for Show is easy
- Need to implement show OR showsPrec
- Let's do show
- Need to go from Pt2 to a String

Assignment Project Exam **Help**

No longer need to derive  
, we've made our own

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

- Use string concatenation to create a pleasing visual output for Pt2
- In doing so, we make use of show as defined for Floats

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Haskell Tutorials/References:

[https://en.wikiboo.org/wiki/Haskell\\_Tutorial](https://en.wikiboo.org/wiki/Haskell_Tutorial)

<https://eduassistpro.github.io/>

<http://cheatsheet.codesio.com/wechat-edu-assist-prosheet.pdf>

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro