

C/CPS 506

Assignment Project Exam Help

Comparative Programming Languages

<https://powcoder.com>

Prof. Alex Ufkes

Add WeChat powcoder

Topic 6: Type systems, pure functional with Haskell

Notice!

**Obligatory copyright notice in the age of digital
delivery and online classrooms:**

Assignment Project Exam Help

<https://powcoder.com>

The copyright to this original work is held by Alex Ufkes. Students registered in course CCPS 505 can use this material 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://powcoder.com>

Add WeChat powcoder
Elixir assignment due March 19

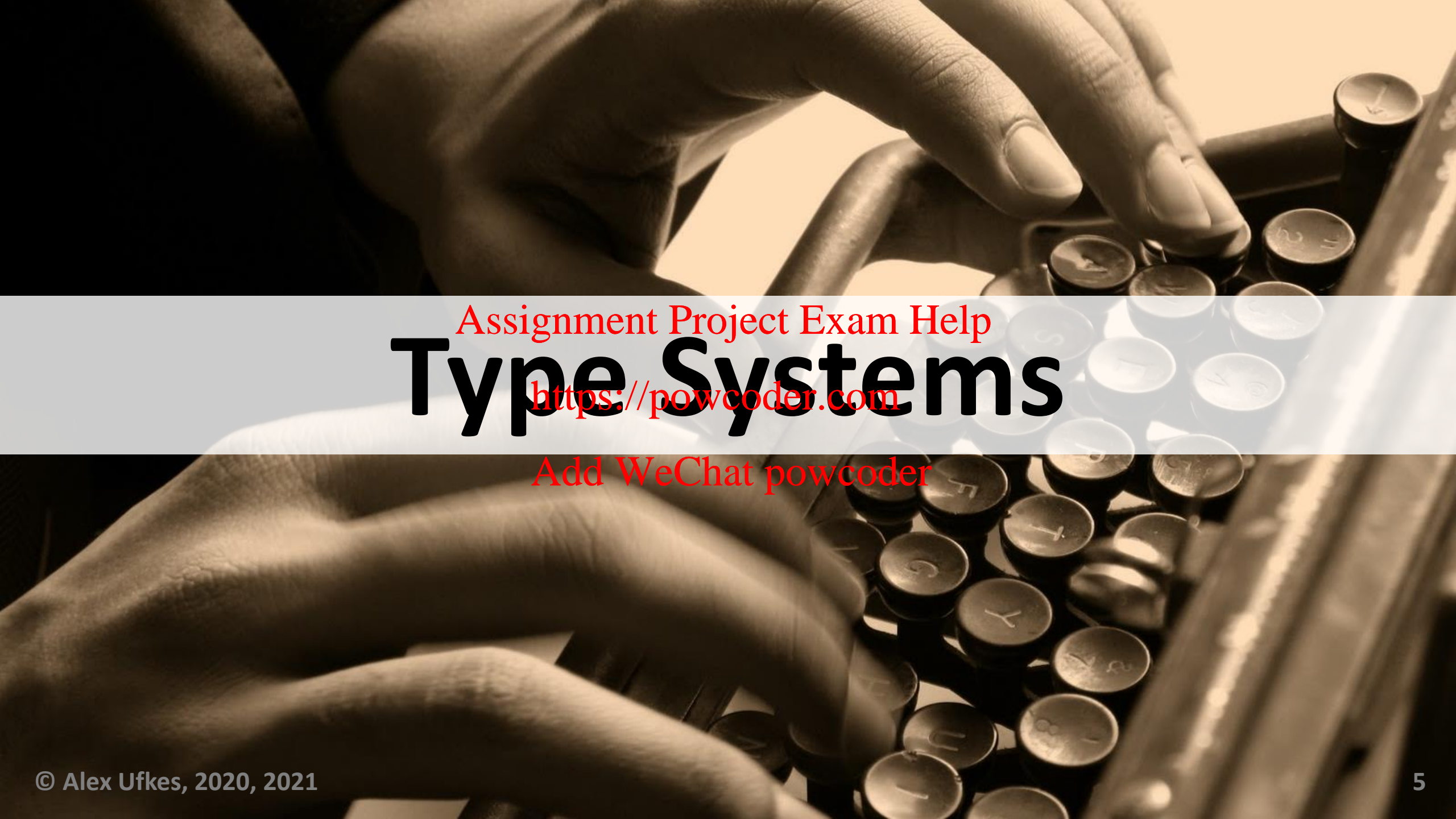
Today

Type systems: Assignment Project Exam Help

- Static VS dynamic
- Strong VS weak

Intro to Haskell Add WeChat powcoder

- Pure functional
- Typing in Haskell



Assignment Project Exam Help

Type Systems

<https://powcoder.com>

Add WeChat powcoder

Type System

- A set of rules that assigns a property called **type** to constructs of a program.
- These constructs include variables, functions, expressions, etc.

<https://powcoder.com>

The whole point is to reduce bugs.

- For example, if a pattern of 32 bits has been encoded using 2s complement, we don't want to read it using IEEE 754
- And we *can* do this in many languages!

```
#include <stdio.h>
#include <windows.h>
```

```
int main(void)
{
```

```
    unsigned long long a = 4607182418800017408;
```

```
    printf("as integer:  %llu\n", a);
    printf("as double:   %lf\n", a);
```

```
    system("pause");
```

```
}
```

Declare large 64-bit integer



Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Print as int, print as double

- The 2s comp bit pattern was read as an IEEE 754 double.
- (The integer constant was deliberately picked to produce a bit pattern that would yield 1.000000 as double)

D:\GoogleDrive\Teaching - Humber\ATMN 253\Visual S

```
as integer:  4607182418800017408
as double:   1.000000
Press any key to continue . . .
```

Type Checking

Clearly, type checking isn't performed in the context of a `printf` statement in C++

Assignment Project Exam Help

- Think of type checking as trying to fit puzzle pieces together.
- Does the output type of a function match the variable we're trying to store it in?
- Do the input arguments to a function match the types indicated in the parameter list?
- If no, will we allow implicit conversion?

Static VS Dynamic

When are types checked?

Statically typed languages perform type checking at *compile time*

- Checked while converting source code to machine (or byte) code

Dynamically typed languages perform type checking at *run-time*

- Checked on the fly while instructions are being executed.

Statically Typed languages: C/C++, Java, Haskell, Rust

Dynamically Typed languages: Python, Smalltalk, Elixir

Static Type Checking

```
public class MethodTester  
{
```

Assignment Project Exam Help

```
    public static void main(String[] args)  
    {
```

<https://powcoder.com>

```
        String s = "Hello";
```

Add WeChat powcoder

```
        System.out.println(Math.sqrt(s));  
    }
```

```
}
```

```
incompatible types: java.lang.String cannot be  
converted to double
```

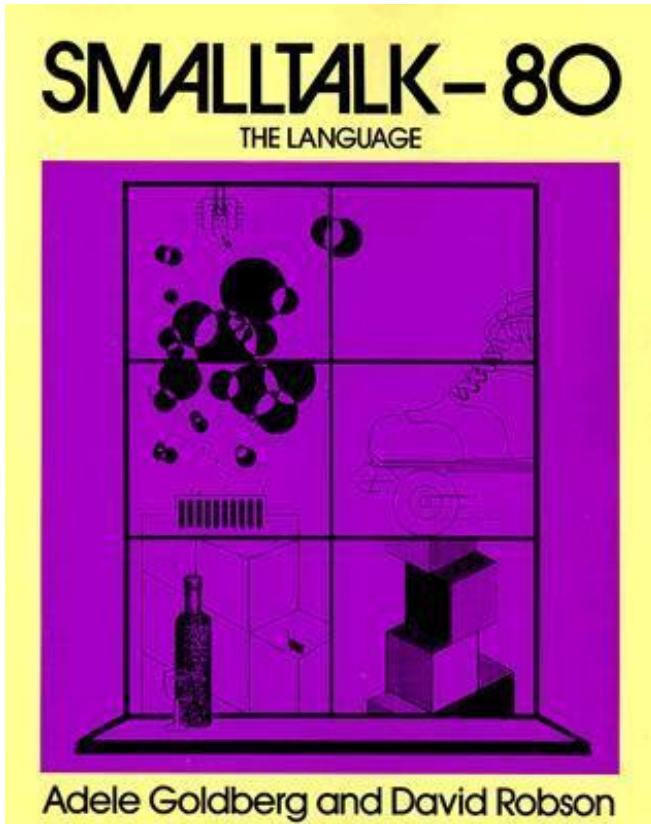
- In dynamically typed languages, every operation knows the types for which it is valid.
- Providing invalid arguments or operands will yield a run-time error which may or may not be recoverable
- Such things can be anticipated and mitigated in various ways, such as verifying type explicitly

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Dynamic Type Checking



```
factorial: n  
  | fac |  
  fac := 1.
```

Assignment Project Exam Help

<https://powcoder.com>

```
n isInteger  
ifTrue: [1 to: n do:  
  [:a | fac := fac*a.].  
  ^fac  
]  
ifFalse: [  
  ^'Bad input'  
].
```

- In Java, the parameter would be defined as **int**
- Compile error if arg isn't **int**, or can't be implicitly cast as an **int**.

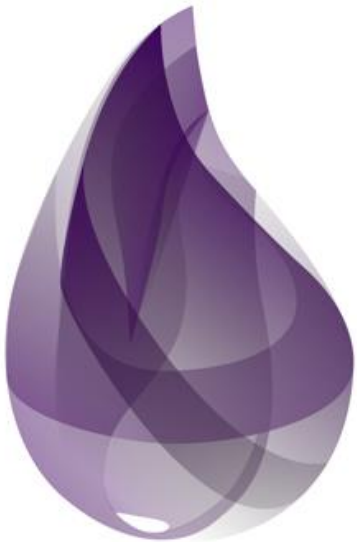
- Of course, polymorphism in Java complicates this.
- Still statically typed.

Dynamic Type Checking...?

`#(1 2 3 4) + 18.2`

- Does Smalltalk have type errors in the strict sense?
- Different objects understand different messages.
- A “type error” occurs when an object doesn’t have a method to handle a particular message.
- “Type” errors in Smalltalk are as a result of not finding a method (DNU, Did Not Understand).
- Above, the error occurs because the Array class doesn’t have an instance *method* for symbolic operator `#+`
- Smalltalk enthusiasts debate this.

Dynamic Type Checking



```
defmodule UserMath do
```

```
  def fib(n) when not is_integer(n) or n < 0 do  
    :error  
  end
```

```
  def fib(0), do: 0  
  def fib(1), do: 1  
  def fib(n), do: fib(n-2) + fib(n-1)
```

```
  def fac(n) when not is_integer(n) or n < 0 do  
    :error  
  end
```

```
  def fac(0), do: 1  
  def fac(n), do: n*fac(n-1)
```

```
end
```

Static VS Dynamic

Advantages? Disadvantages?

Static:

- Reliably find errors at compile time.
- Code will execute faster if types are assumed to be correct at run time.
- Type-specific optimization can be performed at compile time.
- I.e., integer arithmetic is faster than floating point

Dynamic:

- Compilers run faster
- Interpreters can dynamically load new code
 - Smalltalk, MATLAB, iex
- Easier code reuse

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Static VS Dynamic

Advantages? Disadvantages?

- There is much disagreement among programmers about just how much of a problem type errors are in the grand scheme of things.
- Does the added cost of developing in a statically typed language make sense if type-related bugs are but a tiny fraction?
- Of the type-related bugs that occur, what proportion of those would have been solved by a type checker anyway?
- They aren't perfect after all.

Strong VS Weak Typing



Strong VS Weak (or *Loose*)

Refers to how strict statically typed languages are at compile time

There is actually no universally accepted definition of what constitutes strong or weak typing

Assignment Project Exam Help

<https://powcoder.com>

Of strongly typed languages:

Add WeChat powcoder

1974: "Whenever an object is passed from a calling function to a called function, its type must be ***compatible*** with the type declared in the called function."

Compatible is open to interpretation. Is float compatible with double? Integer with short integer?

Strong VS Weak (or *Loose*)

Refers to how strict statically typed languages are at compile time

1974: "Whenever an object is passed from a calling function to a called function, its type must be compatible with the type declared in the called function."

1977: "In a strongly typed language each data area will have a distinct type and each process will state its ***communication requirements*** in terms of these types."

Parameter lists, return types, etc.

Strong VS Weak (or *Loose*)

To what degree does a statically typed language allow implicit type conversion?

The screenshot shows a text editor window titled "Quincy 2005 - [Text1]" containing the following C code:

```
#include <stdio.h>

int main(void)
{
    int y = 3.4;

    printf("%d\n", y);
}
```

Overlaid on the image is a red watermark that reads "Assignment Project Exam Help" and "https://powcoder.com Add WeChat powcoder".

Below the code, a green-bordered box contains the following text:

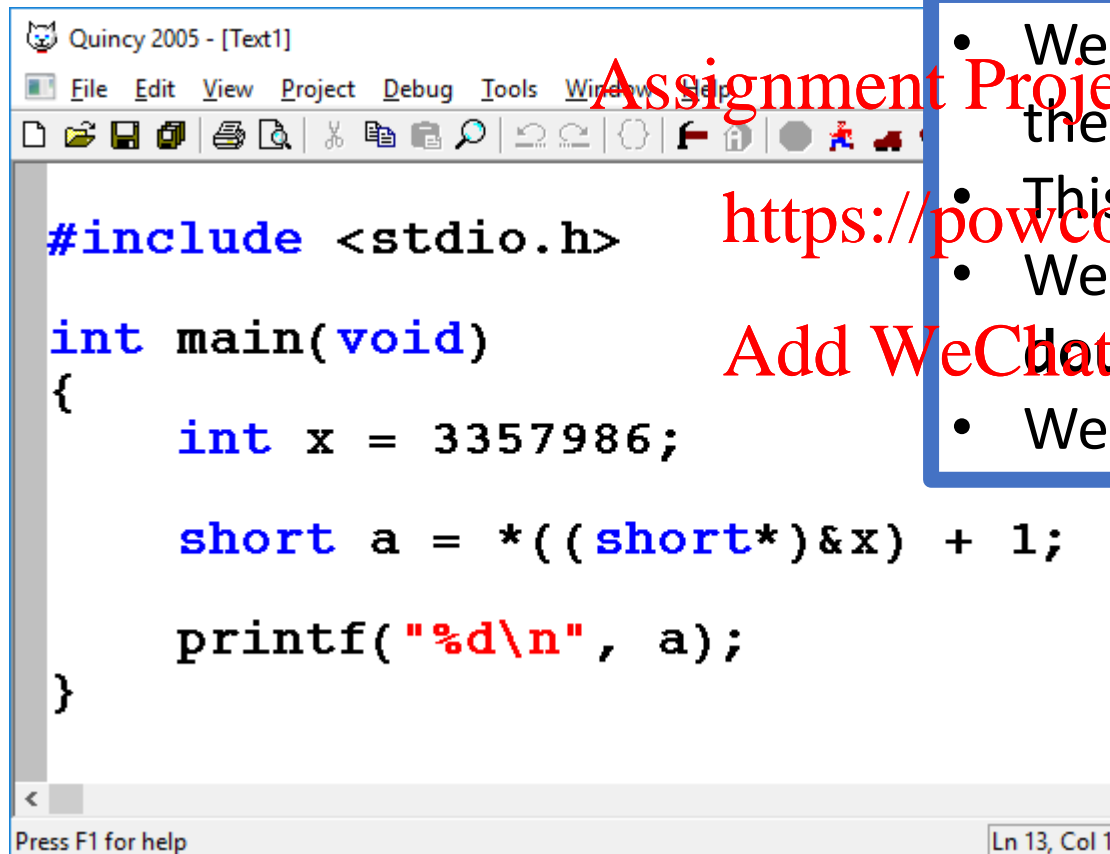
- C is weakly typed.
- Happy to perform all manner of implicit conversion without warning or error.

At the bottom left, there is a copyright notice: "© Alex Lifkes, 2020, 2021".

On the right side of the image, there are two overlapping windows. The top one is a "Build" window showing the output of a build process, which includes the text "Successful build". The bottom one is a terminal window titled "quincy" showing the output of the program: "3", followed by the prompt "Press Enter to return to Quincy...".

Strong VS Weak (or *Loose*)

In C, pointer arithmetic can be used to ***completely bypass*** the type system:



```
Quincy 2005 - [Text1]
File Edit View Project Debug Tools Window Help
#include <stdio.h>

int main(void)
{
    int x = 3357986;

    short a = *((short*)&x) + 1;

    printf("%d\n", a);
}
```

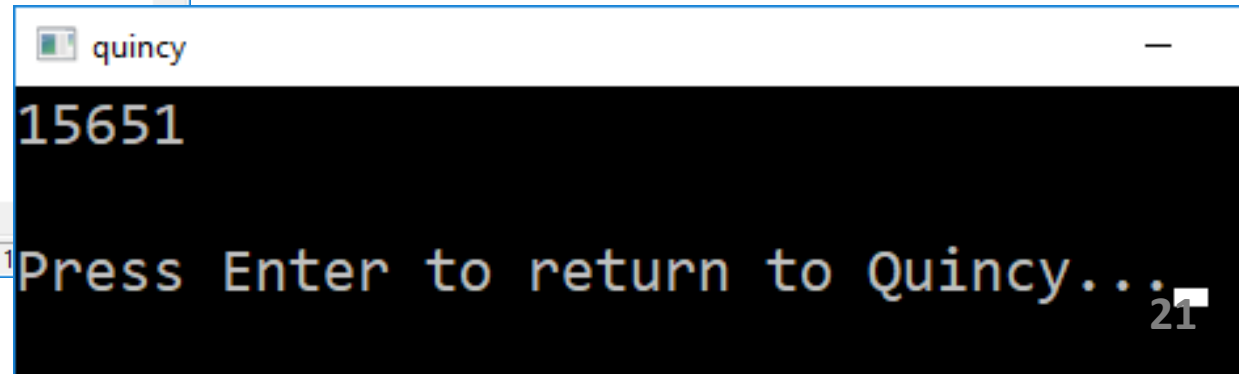
Press F1 for help Ln 13, Col 1

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat: powcoder

- We're using pointer arithmetic to read the first 2 bytes of an **int** as a **short**.
- This can be done with any two types.
- We can read the rightmost 4 bytes of a **double** as an **int**, etc.
- We can treat memory any way we want



```
quincy
15651
Press Enter to return to Quincy...
```



Strong VS Weak (or *Loose*)

C++ will give warnings where C did not, but still compiles and runs in this case:

```
#include <stdio.h>
#include <windows.h>
```

```
int main(void)
{
    int x = 57.99;
```

```
    printf("x = %d\n", x);
```

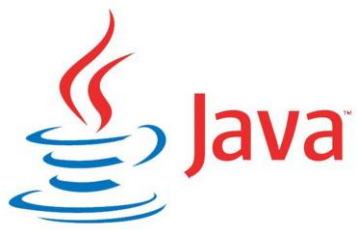
```
    system("pause");
}
```

Configuration: Debug Win32 -----

warning C4244: 'initializing' : conversion from 'double' to 'int', possible loss of data
test\Debug\test.exe
0 to-date, 0 skipped =====

C:\Users\ufkes\Desktop\test\Debug\test.exe

57
Press any key to continue . . .



Strong VS Weak (or *Loose*)

Java will throw compile errors when a *loss of precision* occurs:

```
public class MethodTester
```

```
{
```

```
    public static void main(String[] args)
```

```
    {
```

```
        int a = 7.7;
```

```
        float b = 7.7;
```

```
        float c = 7.7f;
```

```
        double d = 7.7;
```

```
    }
```

```
}
```

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

- No implicit truncation from floating point to integer
- Floating point constants are double precision
- Need to indicate single precision explicitly

Java will throw compile errors when a loss of precision occurs:

Careful! Loss of precision does not **only** occur when going from floating point type to integer type!

Assignment Project Exam Help

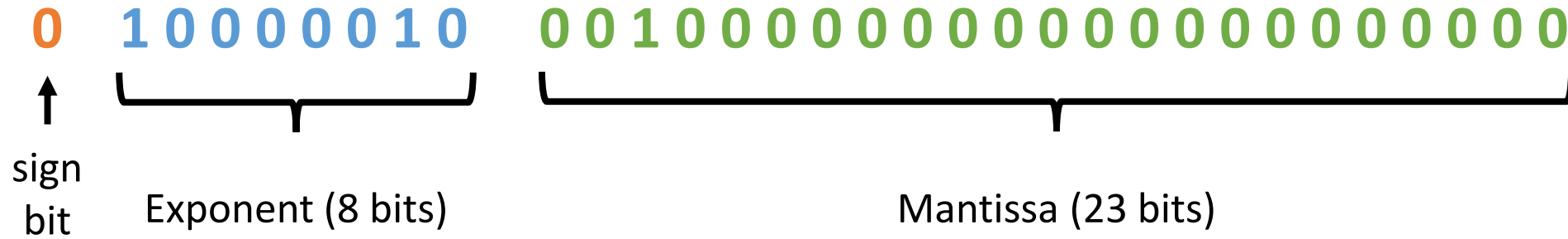
int is 32 bits two's complement.

float is a 23-bit mantissa and an 8 bit exponent.

<https://powcoder.com>

Add WeChat powcoder

32-bit:



Imprecision of Floating Point

- Integers are represented *precisely*. The integer 42 is **exactly** 42.
- The single-precision (32 bits) floating point value 0.1 is **actually** 0.100000001490116119384765625
- *Double*-precision (64 bit) floating point values are more accurate, but still not perfect.

Assignment Project Exam Help

<https://powcoder.com>

But why?

Add WeChat powcoder

- Floating point values exist on an infinite continuum.
- Between any two floating point values are an ***infinite*** number of additional floating point values.
- Integers are discrete. Between any two integers are a ***finite*** number of integers.

Imprecision of Floating Point

- A double-precision float is represented using 64 bits.
- A *finite* number of bits cannot represent an *infinite* number of floating point values.

Assignment Project Exam Help

<https://powcoder.com>

0100101110000010100010100001010100011010101011000110001110000011

Add WeChat powcoder

- There are 2^{64} ways to arrange 64 bits. A large number to be sure, but certainly not infinite.

Infinite Integers?

But there are an infinite number of integers!

- 100% correct. We can't represent every possible integer either.
- Rather, there is a range. A standard 32-bit integer has a range of $-2,147,483,648$ to $2,147,483,647$.
- Every integer within this range is represented precisely.
- Anything outside this range can't be represented using 32 bits
- If we try, we overflow.

```
public class MethodTester
{
    public static void main(String[] args)
    {
        int a = 2111111111;
        System.out.println(a);

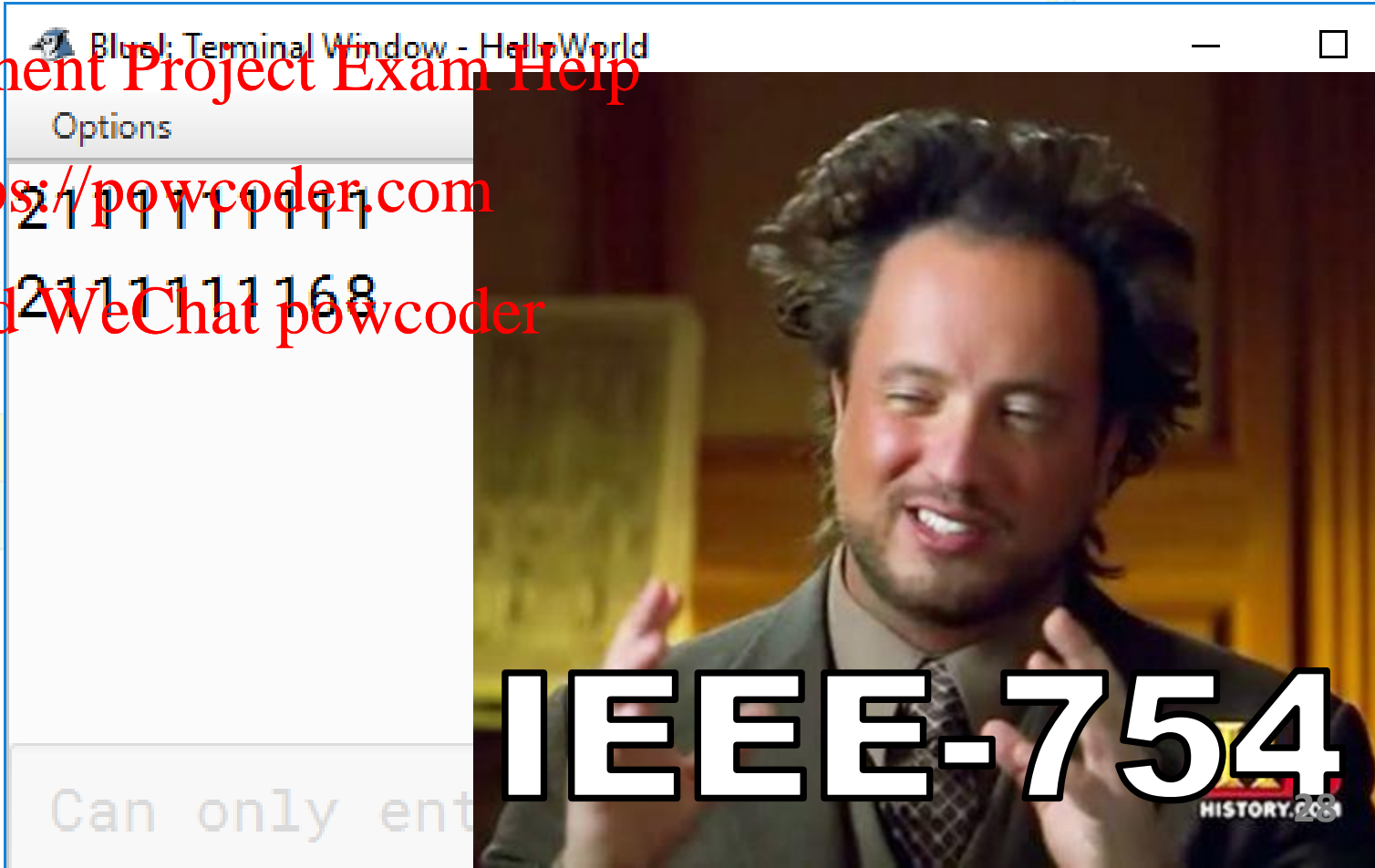
        float b = a;
        a = (int) b;

        System.out.println(a);
    }
}
```

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



A large, stylized 'X' logo composed of four diagonal bars. The left two bars are dark purple, and the right two bars are a lighter, medium purple. The text is overlaid on the center of the 'X'.

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Functional Programming

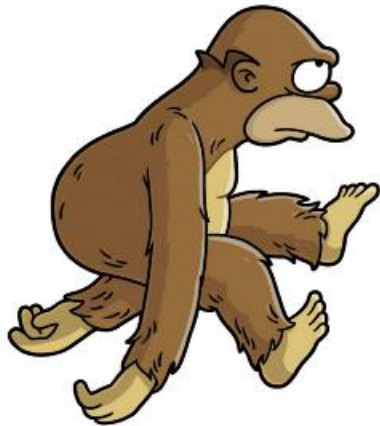
Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



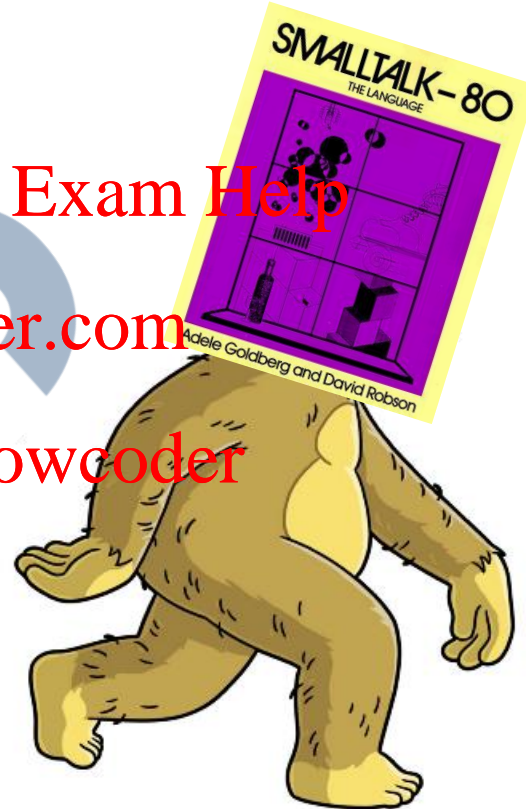
MACHINE



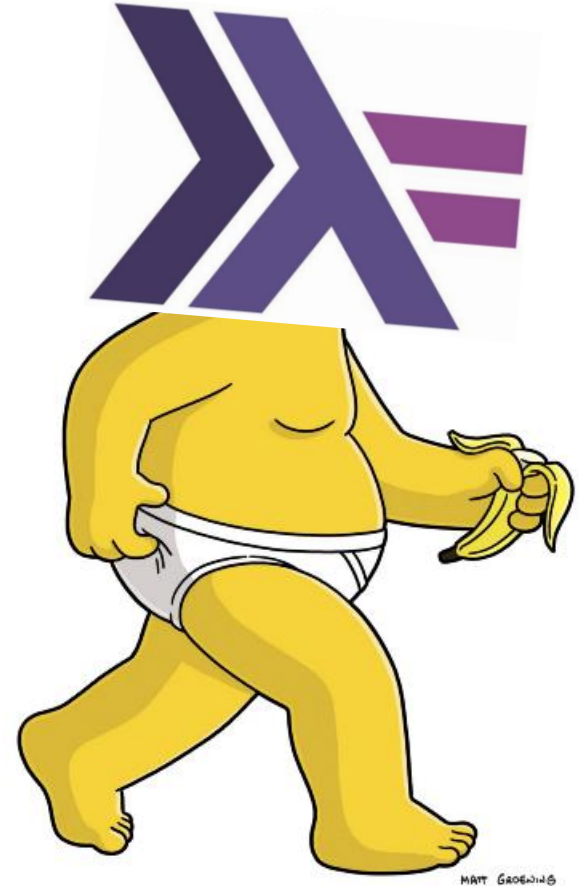
ASSEMBLY



PROCEDURAL

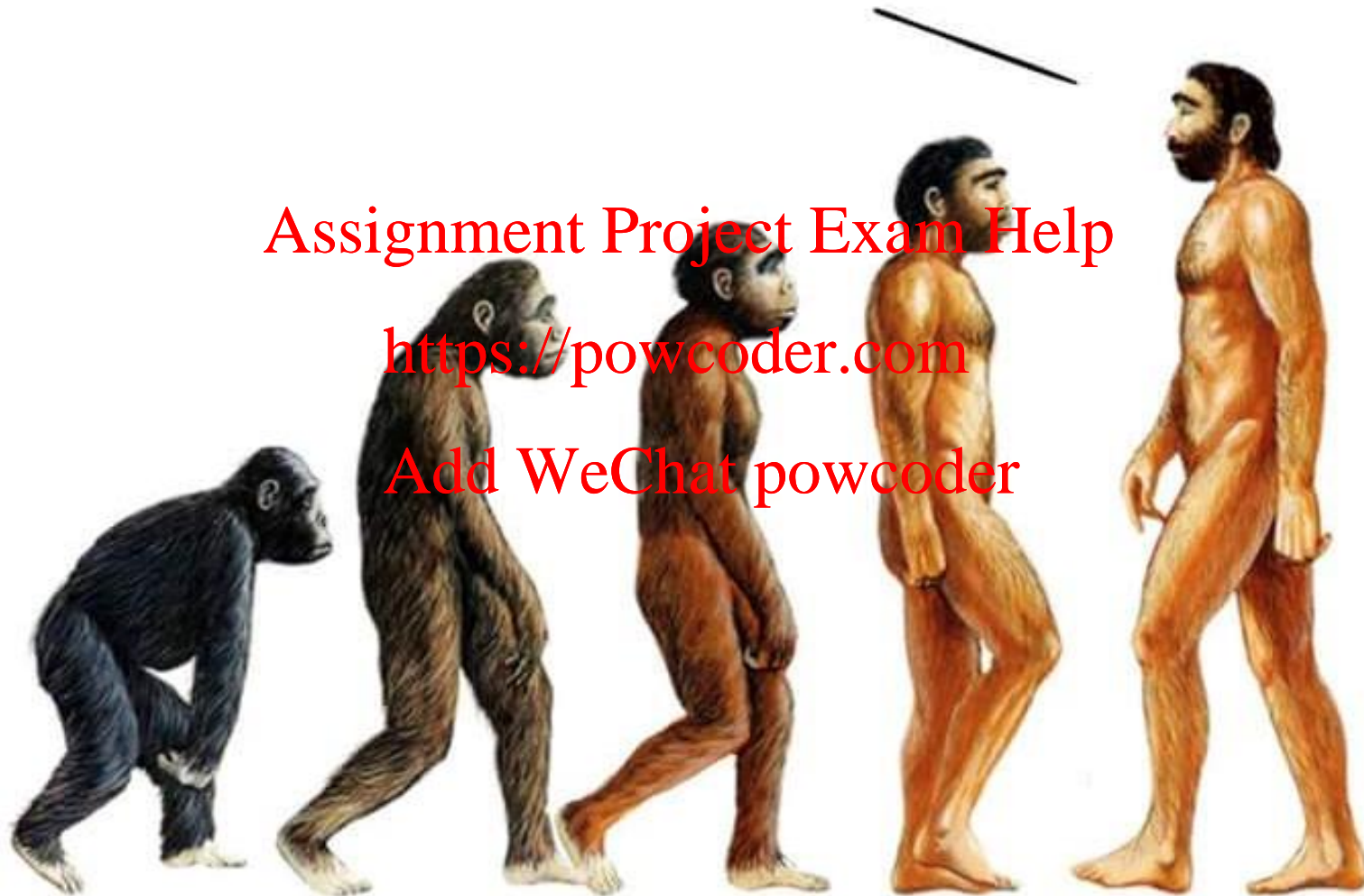


OBJECT ORIENTED



FUNCTIONAL

Go back. We f*cked up.



Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

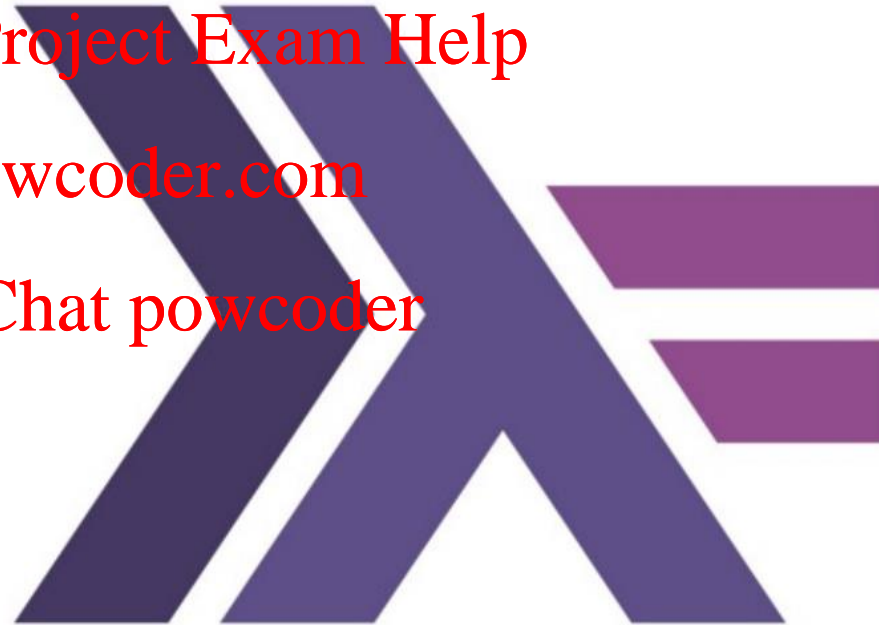
Functional Programming



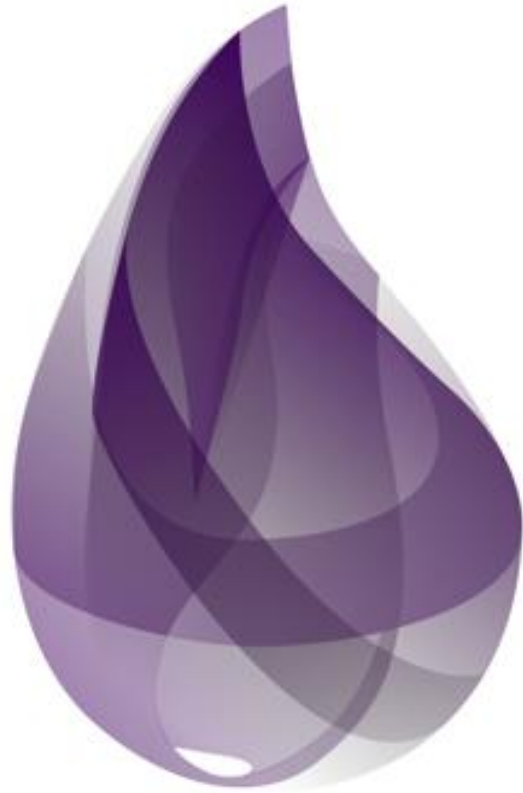
Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



Functional Programming



Higher-order functions:

- Can return functions or accept them as arguments.

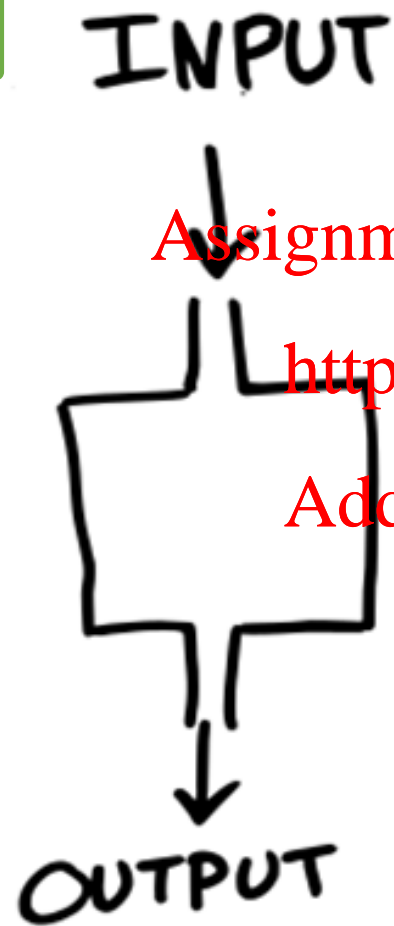
First class functions:

- Can be passed as arguments, returned as values.
- Think of them as **values**, just like integers or floats

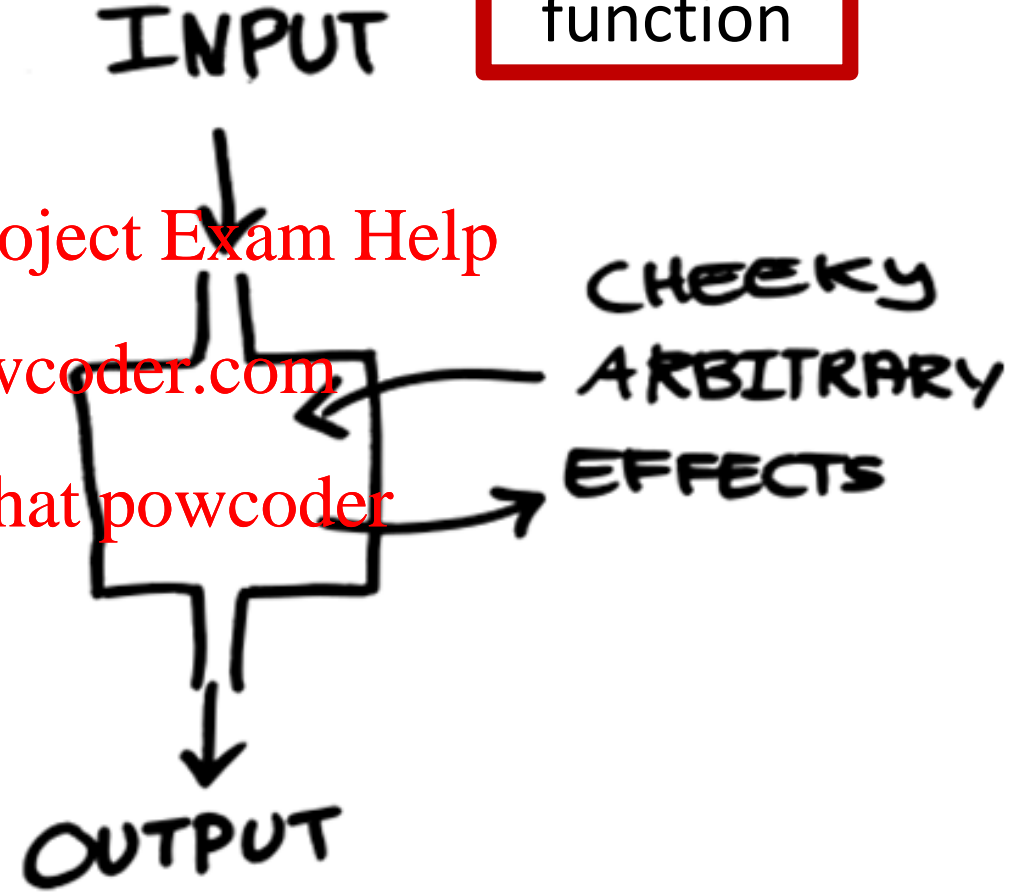
Pure Functions:

- Functions that have no side effects. No interaction with world outside of local scope
- Easier to verify correctness, thread-safe when no data dependency is present.

Pure
function



Impure
function



Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Functional Programming

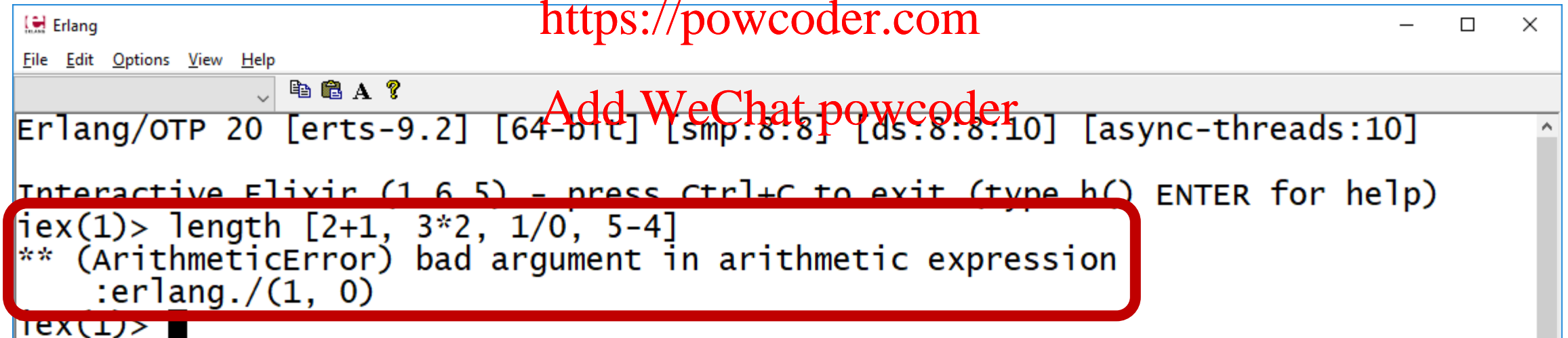
Strict (eager) VS. non-strict (lazy) evaluation:

- Strict: evaluate function arguments before invoking the function.
- Lazy: Evaluates arguments if their value is required to invoke the function.

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

A screenshot of an Erlang shell window. The title bar says 'Erlang'. The menu bar has 'File', 'Edit', 'Options', 'View', and 'Help'. The status bar shows 'Erlang/OTP 20 [erts-9.2] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:10]'. The main text area shows the prompt 'Interactive Elixir (1.6.5) - press Ctrl+C to exit (type h() ENTER for help)'. Below this, the user enters 'iex(1)> length [2+1, 3*2, 1/0, 5-4]'. The shell responds with an error: '** (ArithmeticError) bad argument in arithmetic expression :erlang./(1, 0)'. The error message is highlighted with a red rounded rectangle. The prompt 'iex(1)>' is visible again at the bottom.

```
Erlang/OTP 20 [erts-9.2] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:10]
Interactive Elixir (1.6.5) - press Ctrl+C to exit (type h() ENTER for help)
iex(1)> length [2+1, 3*2, 1/0, 5-4]
** (ArithmeticError) bad argument in arithmetic expression
:erlang./(1, 0)
iex(1)>
```

**Elixir largely performs strict evaluation
(some exceptions, recall Stream, Range)**

Functional Programming

Strict (eager) VS. non-strict (lazy) evaluation:

- Strict: evaluate function arguments before invoking the function.
- Lazy: Evaluates arguments if their value is required to invoke the function.

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Try it!

Type Haskell expressions in here.

```
λ length [2+1, 3*2, 1/0, 5-4]
4 :: Int
λ
```

Got 5 minutes?

Type `help` to start the tutorial.

Or try typing these out and see what happens (click to insert):

```
23 * 36 or reverse "hello" or foldr (:) [] [1,2,3] or do line
<- getLine; putStrLn line or readFile "/welcome"
```

These IO actions are supported in this sandbox.

<https://www.haskell.org/>

A great intro to Haskell syntax

Haskell: Functional Programming cranked up to 11



History



Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

- Named after logician Haskell Curry
- In the late 80s, interest in lazy functional languages was growing
- There was a strong consensus to define an open standard for such languages

History



- Haskell 1.0 was defined in 1990
 - Continued with version 1.1, 1.2, 1.3, etc.
 - Culminated with *Haskell 98*
- Haskell 2010 was published in July 2010
 - Contained uncontroversial features previously enabled via compiler flags
- Next version being worked on – Haskell 2020
 - Though progress seems to have stalled
 - Perhaps it should be called Haskell 202X

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Features



Purely Functional:

- Every function is *pure*
- Even side-effect inducing operations are produced by pure code
- No statements, only expressions
- Cannot mutate variables (local or global)
- Supports pattern matching
- Side effects are handled using *monads*

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Features



Statically Typed:

- Every expression has a type
 - Determined at compile time
- Types composing an expressions must match
 - If not, compile error

Type Inference:

- Types don't have to be written out explicitly
 - Though you can if you want
- They will be inferred at compile time

Features



Lazy Evaluation:

- Functions don't evaluate their arguments
- Control constructs written as functions
- Easy to fuse chains of functions together
- Computation never takes place unless a result is used.

Concurrency:

- GHC (Haskell compiler) includes high performance parallel garbage collector
- Light-weight concurrency library

Haskell in Industry?

Assignment Project Exam Help

<https://powcoder.com>
https://wiki.haskell.org/Haskell_in_industry

Add WeChat powcoder



Haskell has a diverse range of use commercially, from aerospace and defense, to finance, to web startups, hardware design firms and a lawnmower manufacturer. This page collects resources on the industrial use of Haskell.

- The main user conference for industrial Haskell use is CUFP - the [Commercial Users of Functional Programming Workshop](#).
- The [Industrial Haskell Group](#) supports commercial users.
- There is a well-maintained (as of 2018) [github repository](#) that collects information on companies using Haskell.
- The [commercial Haskell group](#) is a special interest group for companies and individuals interested in commercial usage of Haskell.

The Reddit page [72 would-be commercial Haskell users: what Haskell success stories we need to see](#) has several stories of commercial Haskell users.

1 Haskell in Industry

Many companies have used Haskell for a range of projects, including:

- [ABN AMRO](#) Amsterdam, The Netherlands

ABN AMRO is an international bank headquartered in Amsterdam. For its investment banking activities it needs to measure the counterparty risk on portfolios of financial derivatives.

ABN AMRO's [CUFP talk](#).

- Aetion Technologies LLC, Columbus, Ohio

Aetion was a defense contractor in operation from 1999 to 2011, whose applications use artificial intelligence. Rapidly changing priorities make it important to minimize the code impact of changes, which suits Haskell well. Aetion developed three main projects in Haskell, all successful. Haskell's concise code was perhaps most important for rewriting: it made it practicable to throw away old code occasionally. DSELs allowed the AI to be specified very declaratively.

Aetion's [CUFP talk](#).

- Alcatel-Lucent

A consortium of groups, including Alcatel-Lucent, have used Haskell to prototype narrowband software radio systems, running in (soft) real-time.

Notable companies that use or have used Haskell:

- Nvidia
- AT&T
- Ericsson
- Facebook
- Google
- Intel
- Microsoft

Typically Haskell is used on specialized internal projects or research. Not necessarily company-wide.

Installing Haskell:

Assignment Project Exam Help

<https://powcoder.com>
<https://www.haskell.org/>

Add WeChat powcoder



An advanced, purely functional programming language

Declarative, statically typed code.

```
primes = filterPrime [2..]
  where filterPrime (p:xs) =
        p : filterPrime [x | x <- xs, x `mod` p /= 0]
```

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Try it!

Type Haskell expressions in here.

λ

Neat!

Got 5 minutes?

Type `help` to start the tutorial.

Or try typing these out and see what happens (click to insert):

`23 * 36` or `reverse "hello"` or `foldr (:) [] [1,2,3]` or `do line <- getLine; putStrLn line` or `readFile "/welcome"`

These IO actions are supported in this sandbox.



An advanced, purely functional programming language.

Declarative, statically typed code.

```
primes = filterPrime [2..]
where filterPrime (p:xs) =
  p : filterPrime [x | x <- xs, x `mod` p > 0]
```

Haskell Platform

What it is

The Haskell Platform is a self-contained, all-in-one installer. After download, you will have everything necessary to build Haskell programs against a core set of useful libraries. It comes in both minimal versions (with no extra libraries outside of GHC) or full versions, which include a broader set of globally installed libraries.

<https://powcoder.com>

Add WeChat powcoder

What you get

- The Glasgow Haskell Compiler
- The Cabal build system, which can install new packages, and by default fetches from Hackage, the central Haskell package repository.
- the Stack tool for developing projects
- Support for profiling and code coverage analysis
- 35 core & widely-used packages

How to get it

The Platform is provided as a single installer, and can be downloaded at the links below.

- Linux
- OS X
- Windows

Repl.it - CCPS506_Haskell

repl.it/@AlexUfkes/CCPS506Haskell#main.hs

AlexUfkes / CCPS506_Haskell

Files

main.hs

```
1  main = putStrLn "Hello world!"
2
3
4
5
6
7
```

Console

GHCi, version 8.6.5

Hello world!

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

© Alex Ufkes, 2020, 2021

48

Filters

Best match

WinGHCi

Desktop app

Search suggestions

wingh - See web results

wingham

winghouse

winghart's

wingham ontario

winghaven mo

wingham free press

winghaven

winghouse tampa

WinGHCi

File Edit Actions Tools Help

GHCi, version 8.4.2: <http://www.haskell.org/ghc/> :? for help
Prelude> |

Assignment Project Exam Help

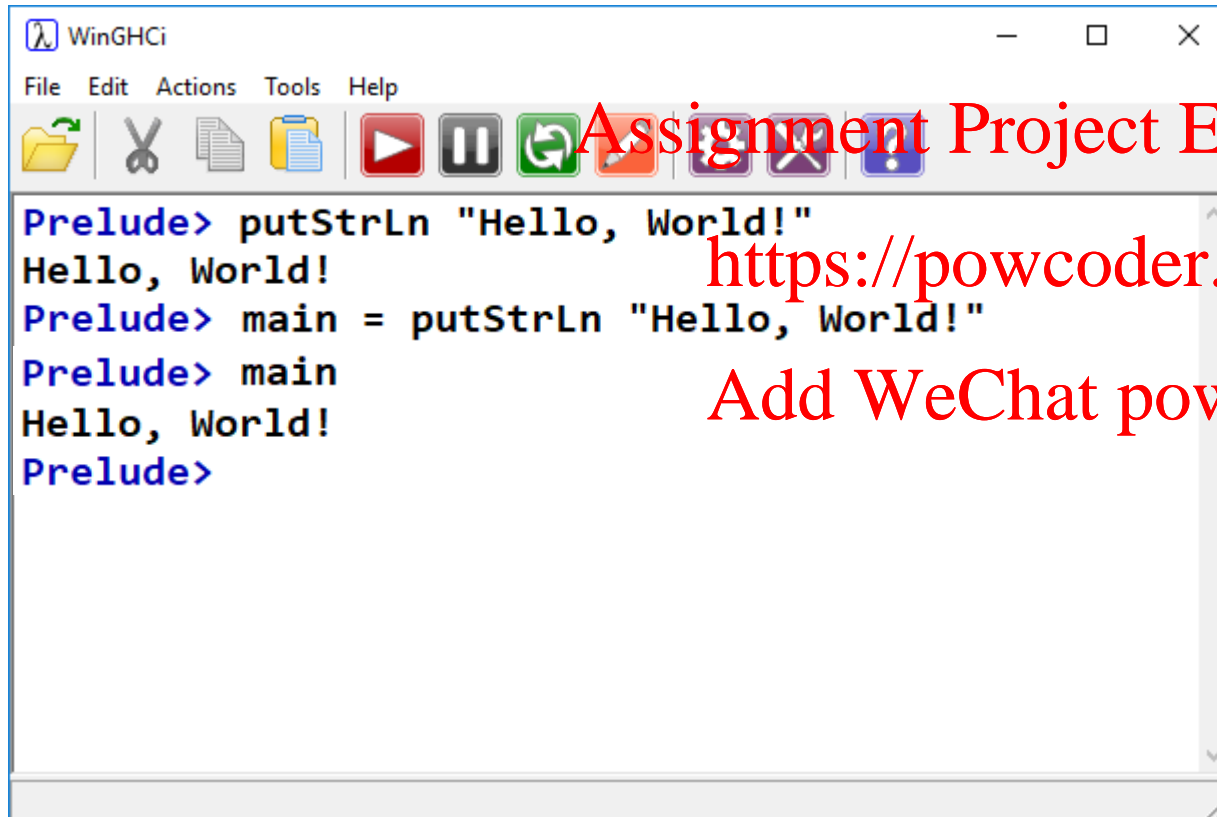
• <https://powcoder.com>

Interactive shell, just like Elixir.

• Haskell's is better!

Add WeChat powcoder

Hello, World!



The screenshot shows a window titled 'WinGHCi' with a menu bar (File, Edit, Actions, Tools, Help) and a toolbar with icons for file operations and execution. The main text area contains the following Haskell code and its output:

```
Prelude> putStrLn "Hello, World!"
Hello, World!
Prelude> main = putStrLn "Hello, World!"
Prelude> main
Hello, World!
Prelude>
```

Assignment Project Exam Help

<https://powcoder.com>

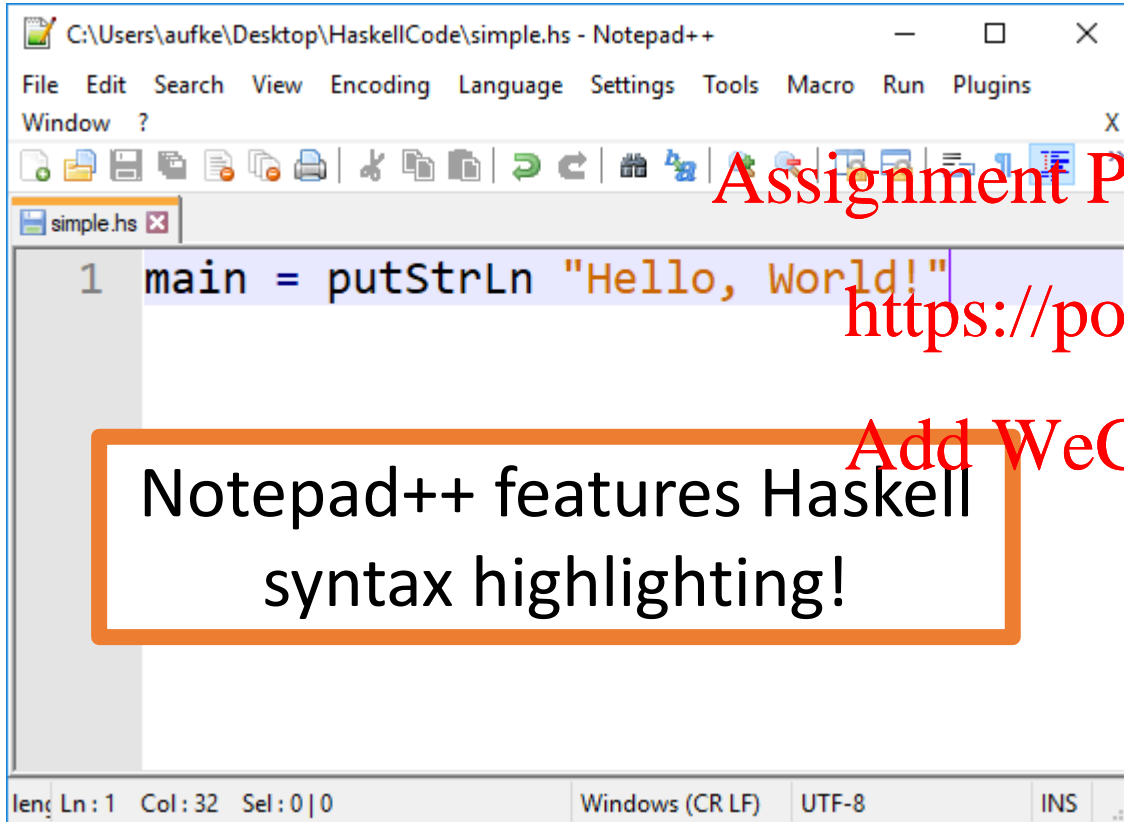
Add WeChat powcoder

```
putStrLn == System.out.println()
putStr == System.out.print()
```

- Define a main function.
- When executing a Haskell program, main is the entry point
- Just like C or Java

Execute main function

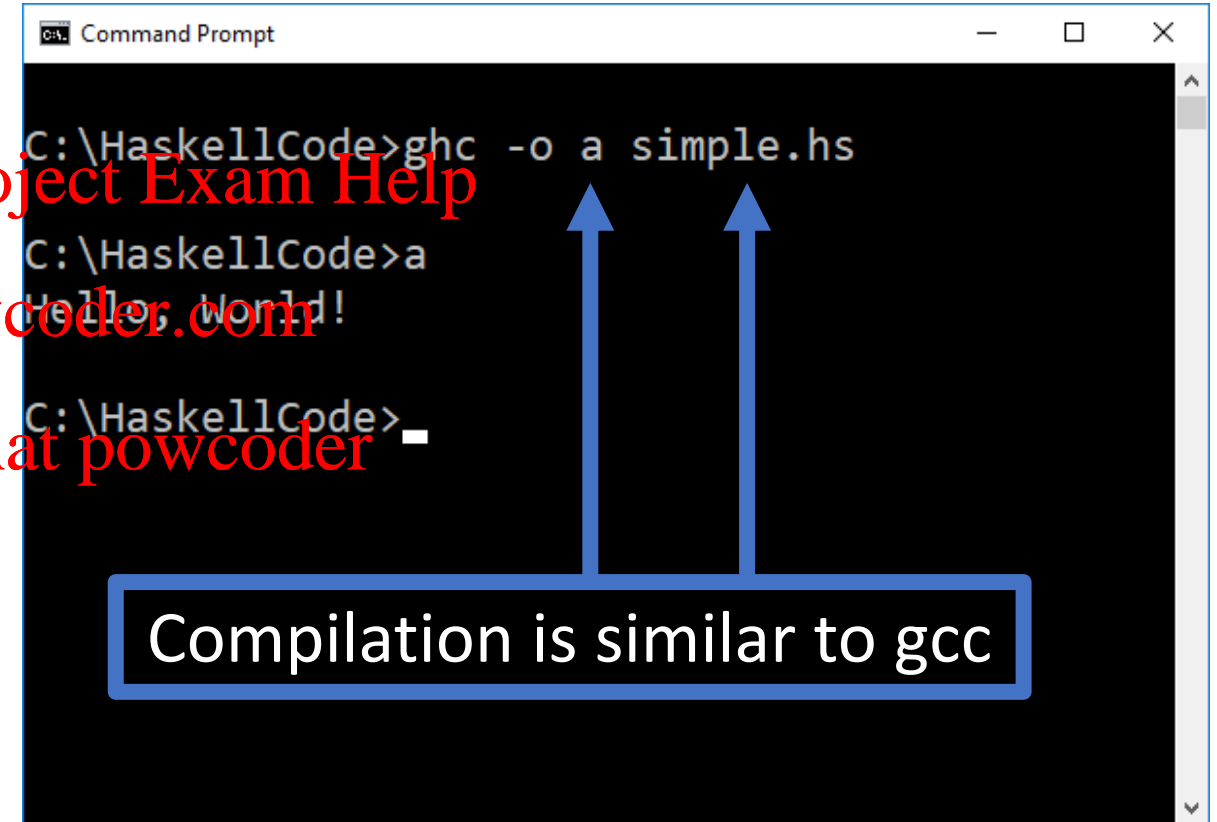
Compiling Haskell



A screenshot of the Notepad++ text editor. The title bar shows the file path: C:\Users\aufke\Desktop\HaskellCode\simple.hs - Notepad++. The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Tools, Macro, Run, and Plugins. The toolbar contains various icons for file operations and editing. The editor window shows a single line of Haskell code: `1 main = putStrLn "Hello, World!"`. The text is color-coded: `main` is blue, `=` is black, `putStrLn` is blue, and the string `"Hello, World!"` is red. A status bar at the bottom shows 'Ln: 1 Col: 32 Sel: 0 | 0', 'Windows (CR LF)', 'UTF-8', and 'INS'.

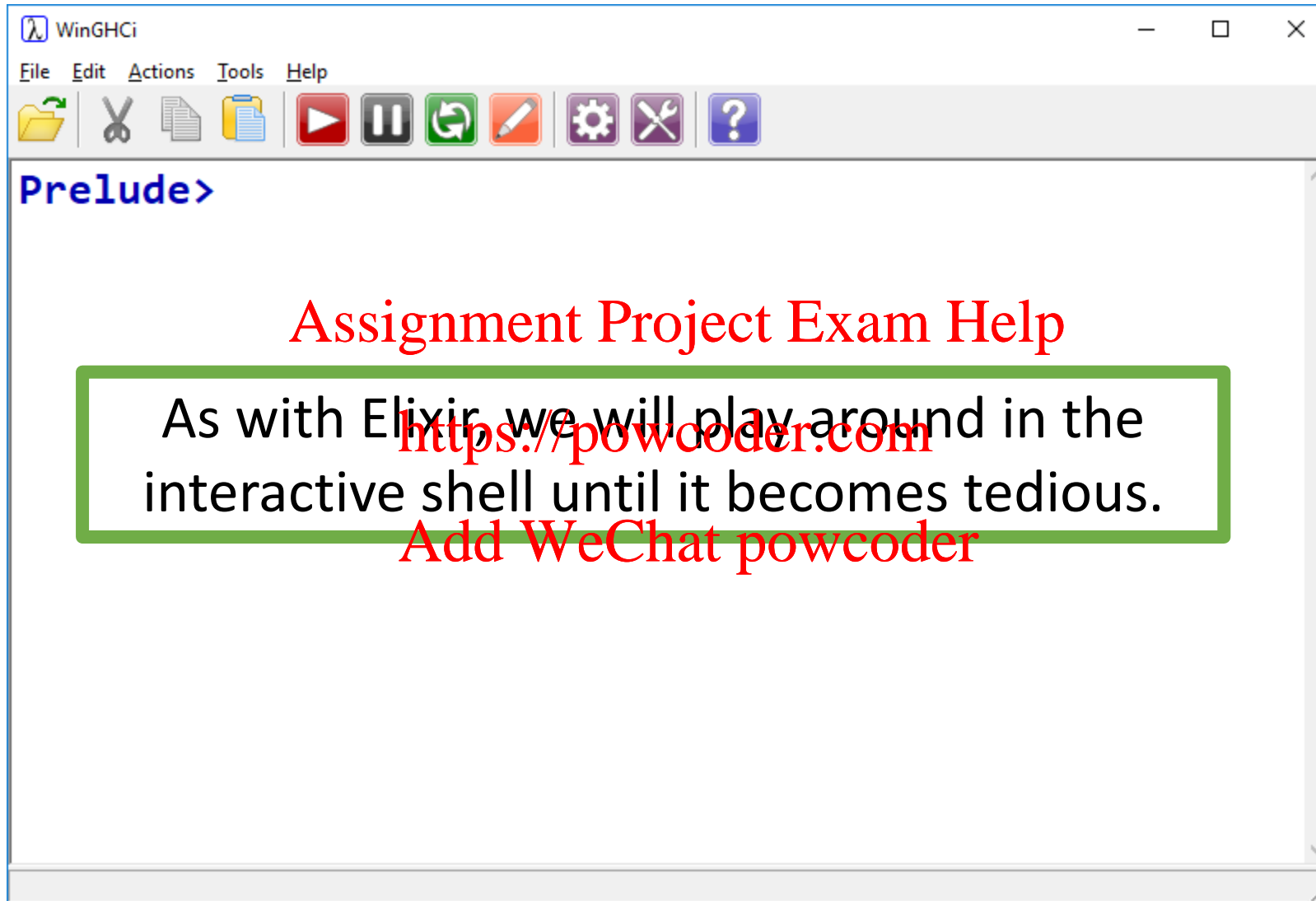
Assignment Project Exam Help
<https://powcoder.com>
Add WeChat powcoder

Notepad++ features Haskell syntax highlighting!

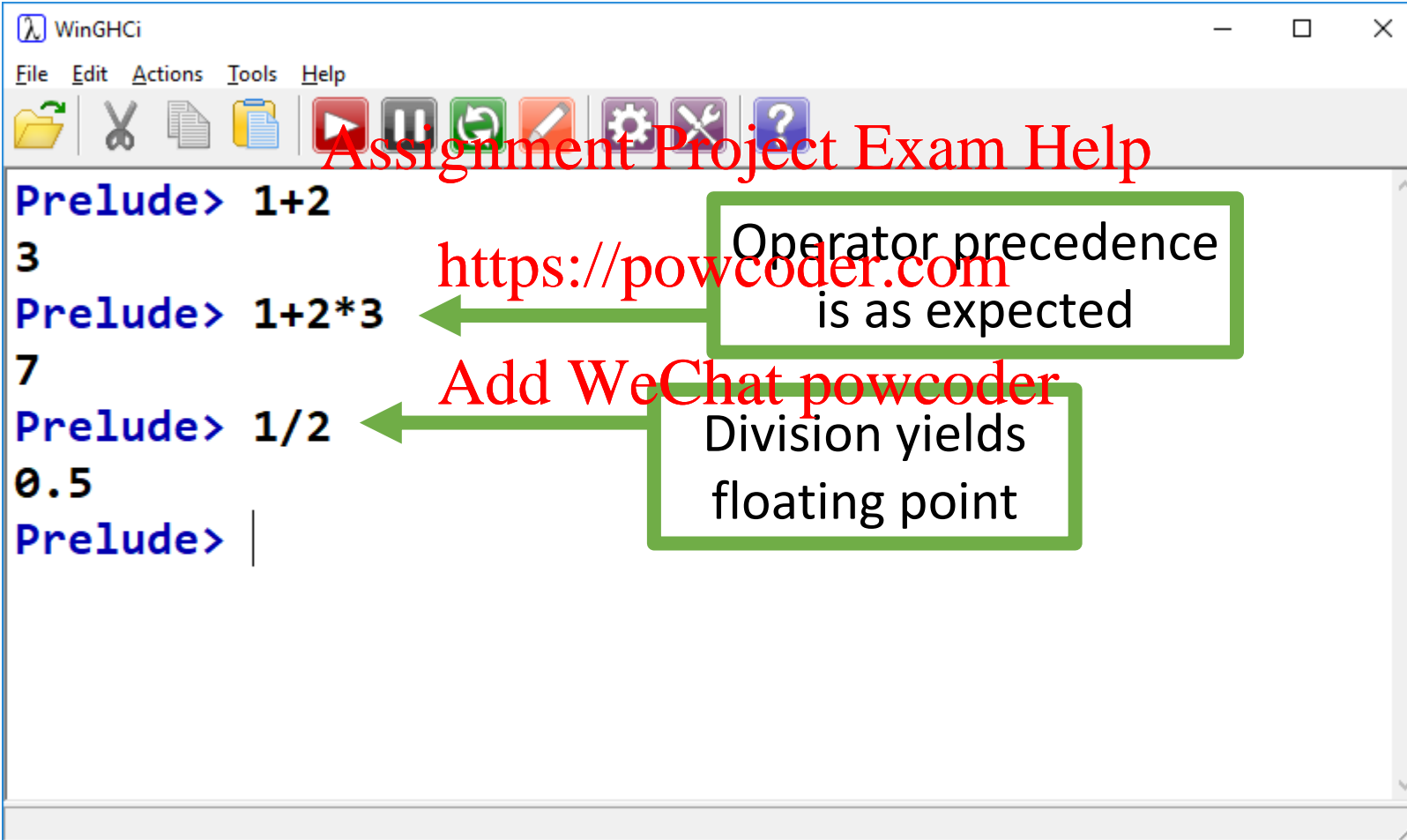


A screenshot of a Windows Command Prompt window. The title bar says 'Command Prompt'. The prompt shows the following commands and output:
`C:\HaskellCode>ghc -o a simple.hs`
`C:\HaskellCode>a`
`Hello, World!`
`C:\HaskellCode>`
Two blue arrows point from a box at the bottom to the `ghc` and `a` commands in the prompt. The box contains the text 'Compilation is similar to gcc'.

Compilation is similar to gcc



Literals & Arithmetic



The screenshot shows the WinGHCi window with the following content:

```
WinGHCi
File Edit Actions Tools Help
[Icons]
Prelude> 1+2
3
Prelude> 1+2*3
7
Prelude> 1/2
0.5
Prelude> |
```

Annotations on the image:

- A red text overlay "Assignment Project Exam Help" is positioned across the top of the window.
- A green box containing the text "Operator precedence is as expected" has a green arrow pointing to the expression `1+2*3`.
- A green box containing the text "Division yields floating point" has a green arrow pointing to the expression `1/2`.
- A red text overlay "Add WeChat powcoder" is positioned between the two green boxes.
- A red text overlay "<https://powcoder.com>" is positioned above the first green box.

Literals & Arithmetic

WinGHCi

File Edit Actions Tools Help

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

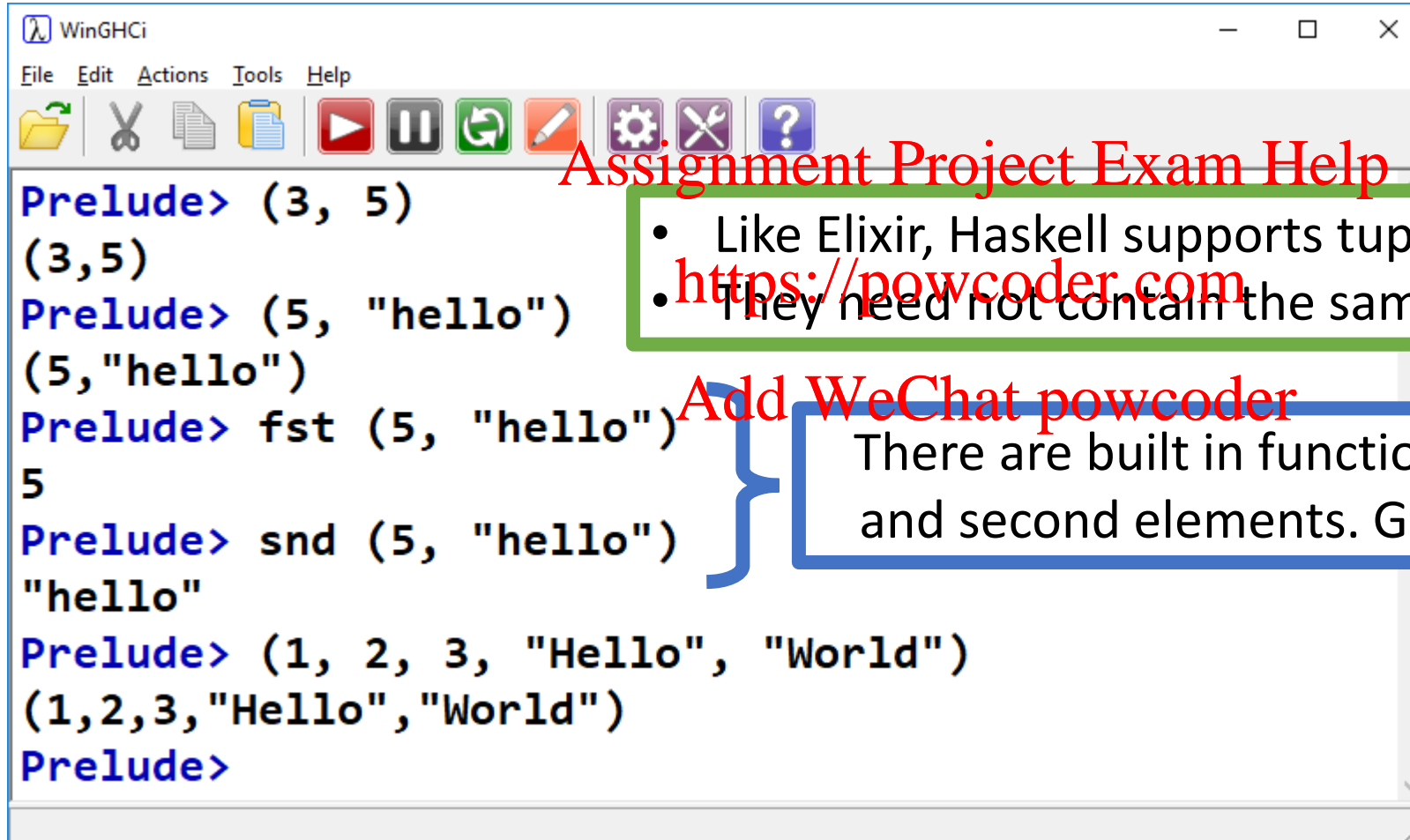
Like Elixir, can omit brackets on function calls

^ can be used for exponentiation

Representation error, no escaping it.

```
Prelude> sqrt 2
1.4142135623730951
Prelude> sqrt (2)
1.4142135623730951
Prelude> 2^2
4
Prelude> sqrt(2)^2
2.0000000000000004
Prelude>
```

Tuples



```
WinGHCi
File Edit Actions Tools Help
[Icons]

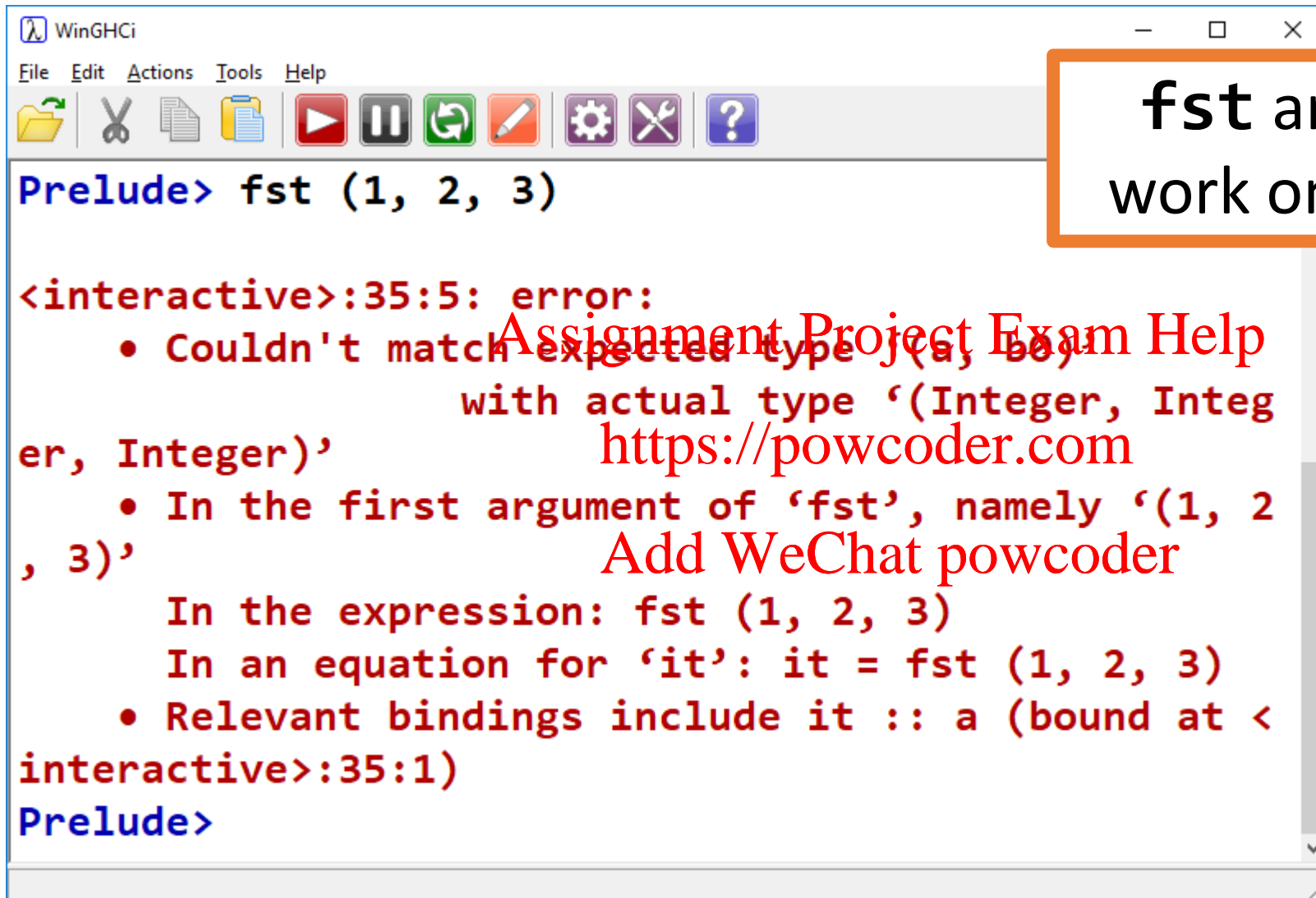
Prelude> (3, 5)
(3,5)
Prelude> (5, "hello")
(5,"hello")
Prelude> fst (5, "hello")
5
Prelude> snd (5, "hello")
"hello"
Prelude> (1, 2, 3, "Hello", "World")
(1,2,3,"Hello","World")
Prelude>
```

Assignment Project Exam Help

- Like Elixir, Haskell supports tuples.
- <https://powcoder.com> They need not contain the same types.

Add WeChat powcoder

There are built in functions for accessing first and second elements. Great for coordinates.



```
WinGHCi
File Edit Actions Tools Help
[Icons]
Prelude> fst (1, 2, 3)

<interactive>:35:5: error:
  • Couldn't match expected type '(a, b)'
    with actual type '(Integer, Integer, Integer)'
  • In the first argument of 'fst', namely '(1, 2, 3)'
    In the expression: fst (1, 2, 3)
    In an equation for 'it': it = fst (1, 2, 3)
  • Relevant bindings include it :: a (bound at <interactive>:35:1)
Prelude>
```

fst and **snd** only
work on pair tuples!

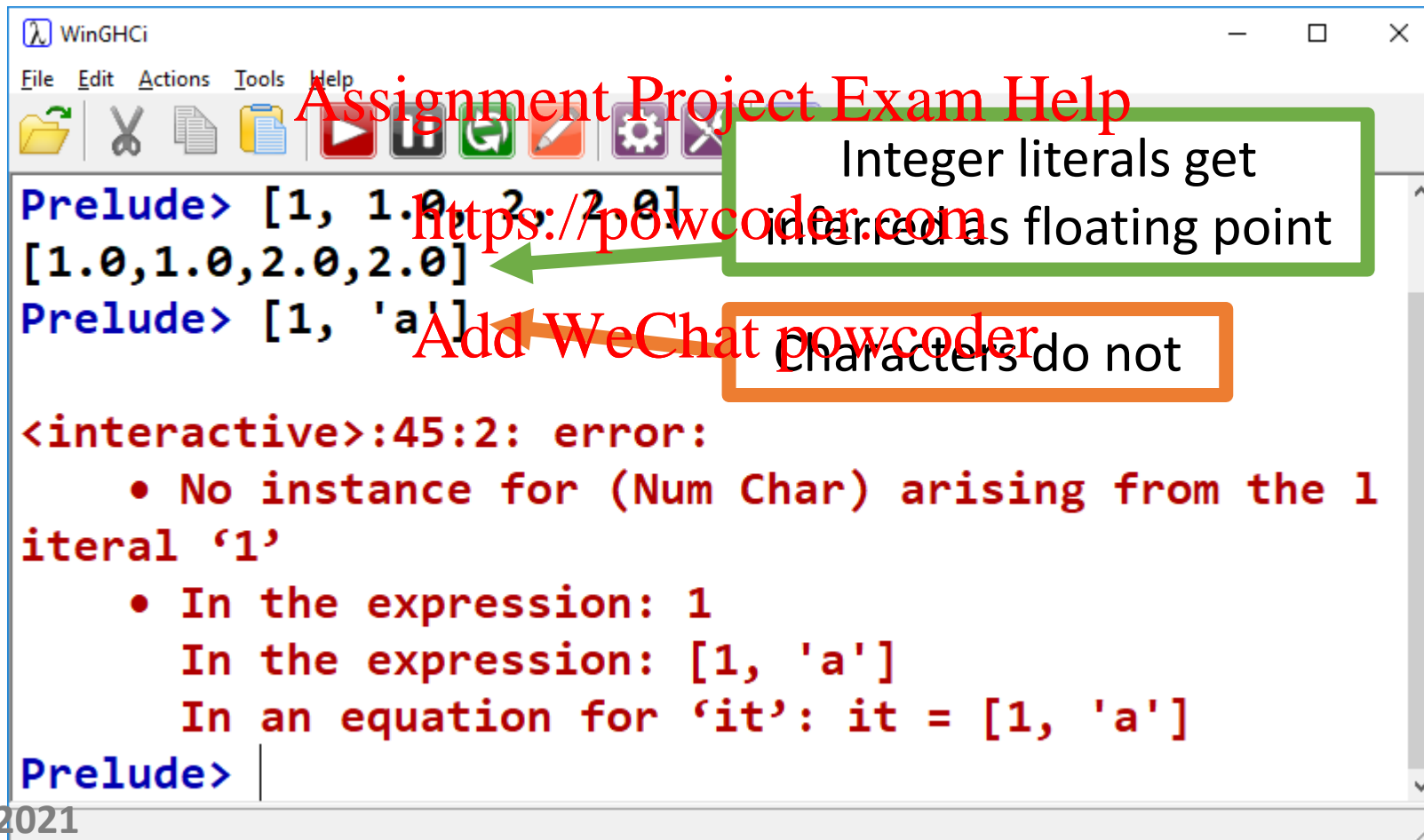
Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Lists

Must be *homogeneous*:



Assignment Project Exam Help
<https://powcoder.com>
Add WeChat powcoder

```
WinGHCi
File Edit Actions Tools Help
[1, 1.0, 2, 2.0]
[1.0, 1.0, 2.0, 2.0]
Prelude> [1, 'a']

Characters do not

<interactive>:45:2: error:
  • No instance for (Num Char) arising from the 1
  literal '1'
  • In the expression: 1
    In the expression: [1, 'a']
    In an equation for 'it': it = [1, 'a']
Prelude>
```

Integer literals get inferred as floating point

Characters do not

Lists

Elements can be added to the *beginning* of a list with the **cons** (:) operator

```
WinGHCi
File Edit Actions Tools Help
[Icons]
Prelude> 0:[1, 2]
[0,1,2]
Prelude> 0:1:2:3:4:[ ]
[0,1,2,3,4]
Prelude> |
```

Assignment Project Exam Help

<https://powcoder.com>

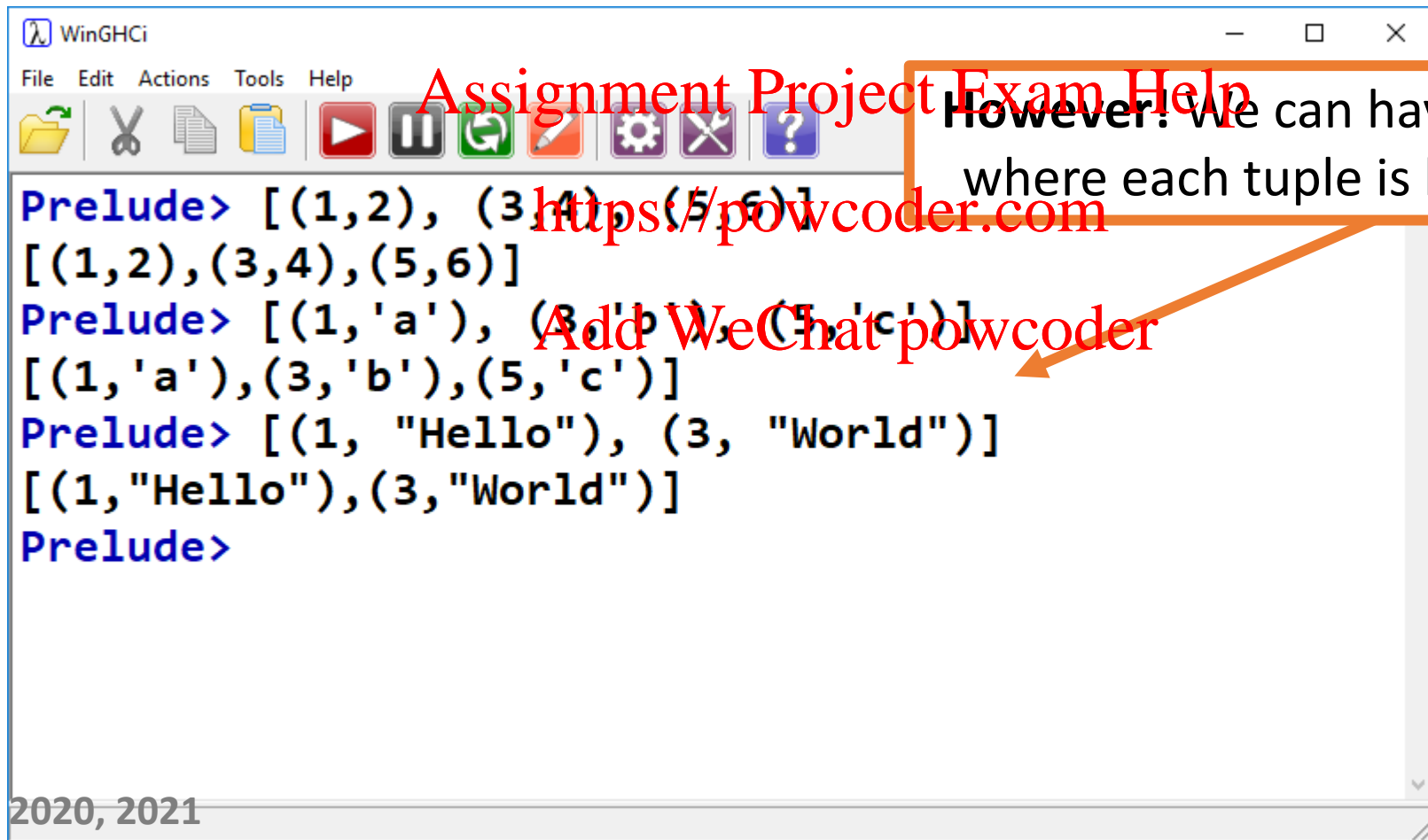
Add WeChat powcoder

Build a list using cons operator and an empty list

In fact, when we write `[1, 2, 3]` the compiler is actually doing `1:2:3:[]`
`[1, 2, 3]` notation is *syntactic sugar*.

Lists & Tuples

Tuples can be heterogeneous, lists must be homogeneous.



```
WinGHCi
File Edit Actions Tools Help
[Icons]
Prelude> [(1,2), (3,4), (5,6)]
[(1,2),(3,4),(5,6)]
Prelude> [(1,'a'), (3,'b'), (5,'c')]
[(1,'a'),(3,'b'),(5,'c')]
Prelude> [(1, "Hello"), (3, "World")]
[(1,"Hello"),(3,"World")]
Prelude>
```

However: We can have lists of tuples, where each tuple is heterogeneous.

Lists & Tuples

Tuples can be heterogeneous, lists must be homogeneous.

```
WinGHCi
File Edit Actions Tools Help
Prelude> [(1, "Hello"), (2, "World")]
[(1, "Hello"), (2, "World")]
Prelude> [(1, "Hello"), (2, 3.4)]
<interactive>:67:20: error:
• Could not deduce (Fractional [Char])
  arising from the literal '3.4'
  from the context: Num a
    bound by the inferred type of it :: Num a => [(a, [Char])]
    at <interactive>:67:1-24
• In the expression: 3.4
  In the expression: (2, 3.4)
  In the expression: [(1, "Hello"), (2, 3.4)]
Prelude> |
```

Assignment Project Exam Help

<https://powcoder.com>

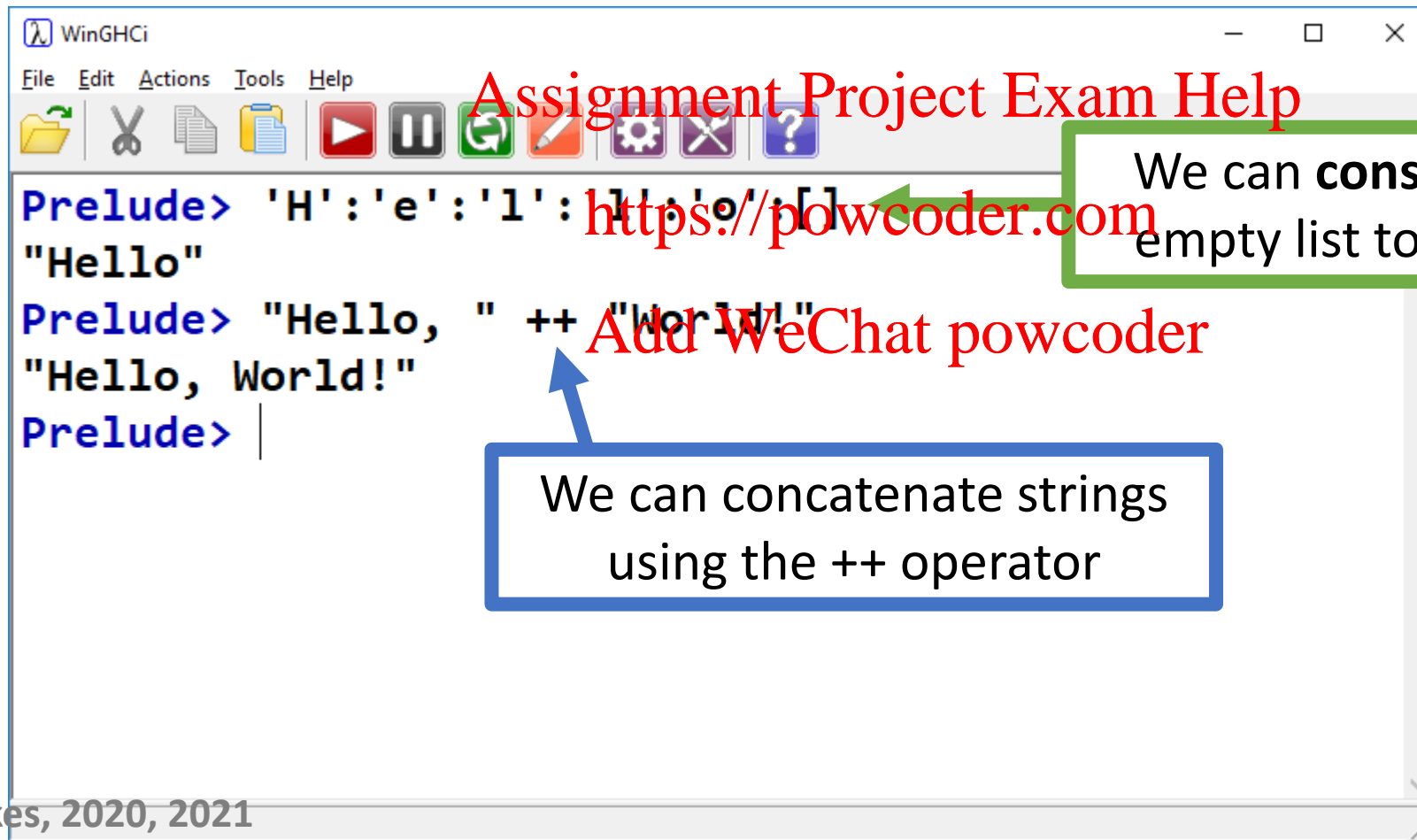
Add WeChat powcoder

However #2:

- In a list of tuples, each tuple must have the same format:

Strings

Strings are simply lists of chars:



The screenshot shows the WinGHCi window with a menu bar (File, Edit, Actions, Tools, Help) and a toolbar. The command line shows the following interactions:

```
Prelude> 'H':'e':'l':'l':'o':[]  
"Hello"  
Prelude> "Hello, " ++ "World!"  
"Hello, World!"  
Prelude> |
```

Annotations on the image include:

- A red watermark "Assignment Project Exam Help" and the URL "https://powcoder.com" are overlaid on the top half of the window.
- A green box on the right contains the text: "We can **cons** chars into an empty list to form a string", with a green arrow pointing to the first command.
- A blue box at the bottom contains the text: "We can concatenate strings using the ++ operator", with a blue arrow pointing to the second command.
- Red text "Add WeChat powcoder" is positioned between the two boxes.

Strings

Concatenate multiple types? Java lets us...

Assignment Project Exam Help
<https://powcoder.com>
Add WeChat powcoder

```
StringTester - HelloWorld
Class Edit Tools
StringTester X
Compile Undo Cut

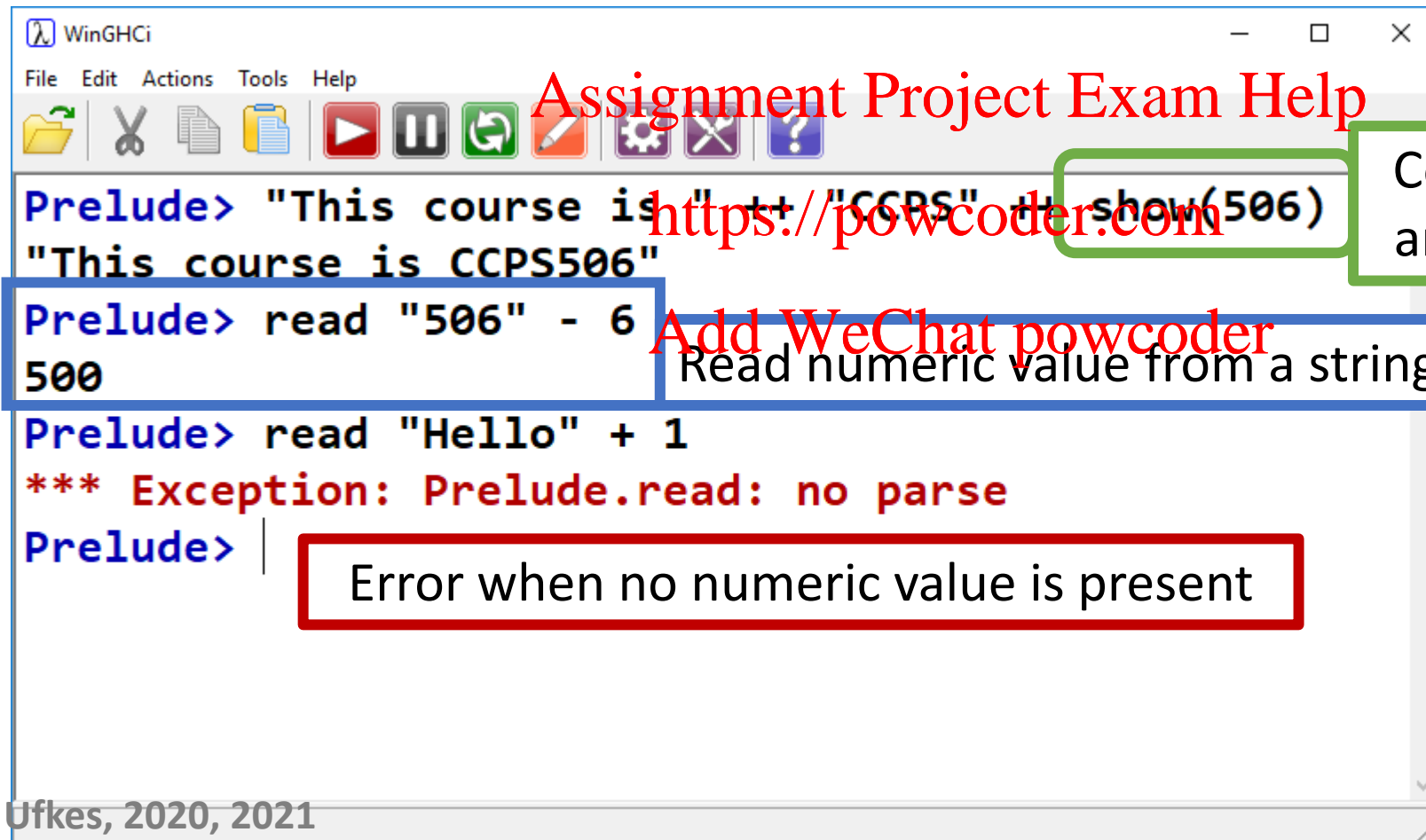
public class StringTester {
    public static void main(String[] args) {
        String s = "Hello, " ++ 5.0
    }
}
```

WinGHCi
File Edit Actions Tools Help
Prelude> "Hello, " ++ 5.0
<interactive>:61:14: error:
• No instance for (Fractional [Char])
 arising from the literal '5.0'
• In the second argument of '(++)', namely '5.0'
In the expression: "Hello, " ++ 5.0
In an equation for 'it': it = "Hello, " ++ 5.0
Prelude>

put wh

Strings

`show()` and `read()` functions



A screenshot of a WinGHCi terminal window. The window has a menu bar with 'File', 'Edit', 'Actions', 'Tools', and 'Help'. Below the menu bar is a toolbar with icons for file operations and execution. The terminal shows the following interactions:

```
Prelude> "This course is " ++ "CCPS" ++ show(506)
"This course is CCPS506"
Prelude> read "506" - 6
500
Prelude> read "Hello" + 1
*** Exception: Prelude.read: no parse
Prelude> |
```

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Convert non-string argument to string

Read numeric value from a string (like *sscanf* in C)

Error when no numeric value is present

Operations on Lists

- In functional programming, computation is done in large part by operating on lists.
- We saw the `hd`, `tl`, `|`, and `Enum` in Elixir.
- Haskell has a similar set of operations.

Add WeChat powcoder

Three primary list-processing functions: `map`, `filter`, `foldr` (and `foldl`)

Head & Tail

```
WinGHCi
File Edit Actions Tools Help
[Paste] [Cut] [Copy] [Run] [Stop] [Refresh] [Undo] [Redo] [Settings] [Tools] [Help]

Prelude> tail [1, 2, 3]
[2,3]
Prelude> head [1, 2, 3]
1
Prelude> tail [1]
[]
Prelude> head [1]
1
Prelude> tail []
*** Exception: Prelude.tail: empty list
Prelude>
```

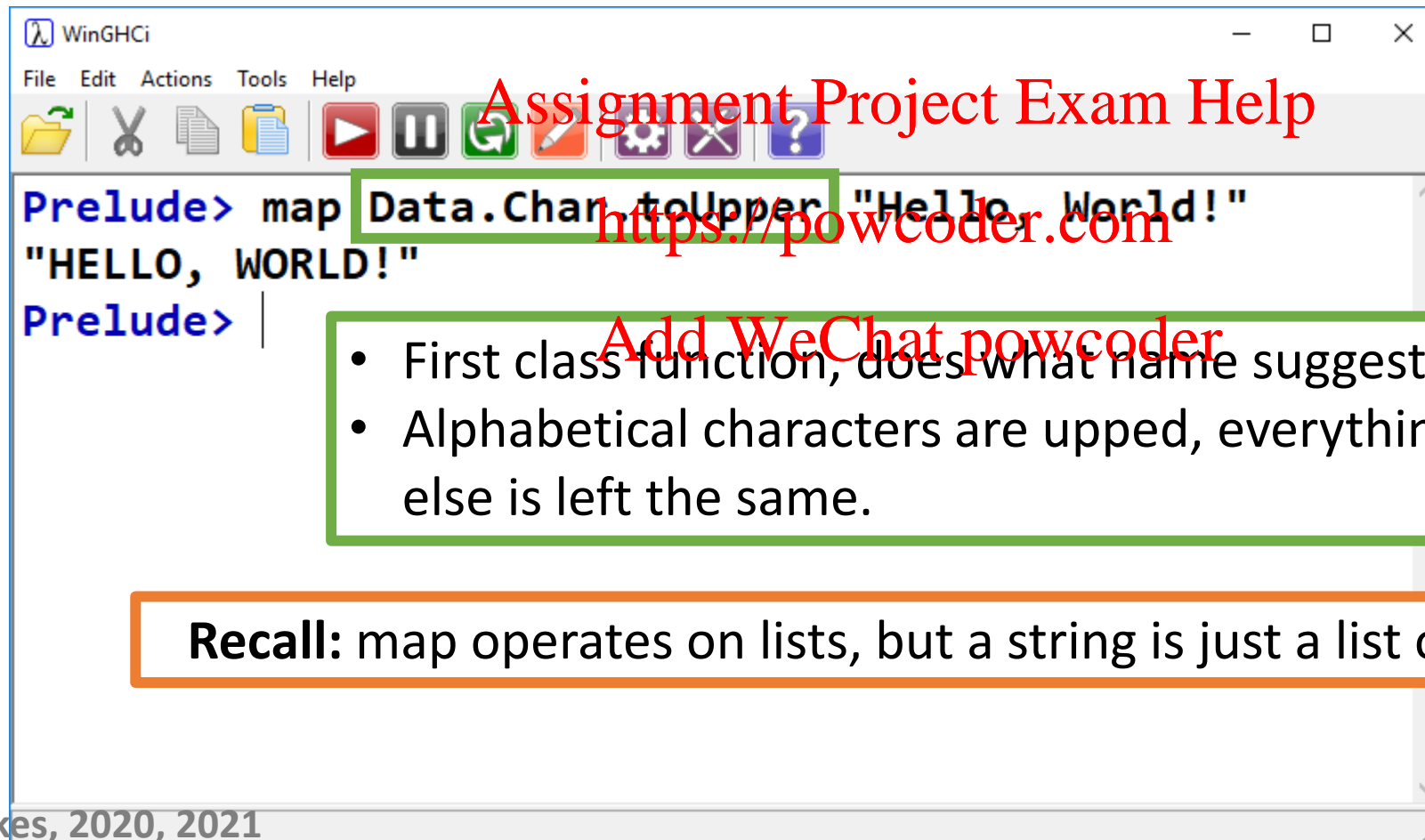
Assignment Project Exam Help
<https://powcoder.com>
Add WeChat powcoder

Same as Elixir:

- Head returns the first element
- Tail returns the rest, as a list
- Note boundary cases:
 - Single element lists
 - Empty lists

map

Similar to Elixir's `Enum.map`



The screenshot shows the WinGHCi window with a menu bar (File, Edit, Actions, Tools, Help) and a toolbar. The command prompt shows the following interaction:

```
Prelude> map Data.Char.toUpper "Hello, World!"  
"HELLO, WORLD!"  
Prelude> |
```

Annotations on the image include:

- A red watermark "Assignment Project Exam Help" and the URL "https://powcoder.com" are overlaid on the top right of the terminal window.
- A green box highlights the `Data.Char.toUpper` function in the command.
- A green box contains the text "Add WeChat powcoder" and a bulleted list:
 - First class function, does what name suggests.
 - Alphabetical characters are upped, everything else is left the same.
- An orange box at the bottom contains the text: **Recall:** map operates on lists, but a string is just a list of characters

map

Similar to Elixir's `Enum.map`

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

```
WinGHCi
File Edit Actions Tools Help
Prelude> map Data.Char.toUpper "Hello, World!"
"HELLO, WORLD!"
Prelude> |
```

Map takes two arguments: A function, and a list of values to which the function is to be applied.

filter

“Remove” items from a list based on some criteria:

```
WinGHCi
File Edit Actions Tools Help
[Paste] [Cut] [Copy] [Undo] [Redo] [Find] [Run] [Exit] [Help]

Prelude> map Data.Char.toUpper "Hello, World!"
"HELLO, WORLD!"
Prelude> filter Data.Char.isLower "Hello, World!"
"elloorld"
Prelude> |
```

Assignment Project Exam Help
<https://powcoder.com>
Add WeChat powcoder

Function List

foldl, foldr

Replaces the cons operator with some other function. This takes some explaining.

Assignment Project Exam Help

Recall that the list:

<https://powcoder.com>
[1, 2, 3, 4, 5]

Add WeChat powcoder

Is actually seen as:

1:2:3:4:5:[]

By the compiler.

foldl, foldr

Replaces the cons operator with some other function. This takes some explaining.

Recall that the list:

[1, 2, 3, 4, 5]

Is actually seen as:

1:2:3:4:5:[]

By the compiler.

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

- **foldr** in effect replaces the cons operator with another function of our choosing.
- This is similar to **Enum.reduce** in Elixir.
- The empty list is replaced with some initial value.

foldl, foldr

Replaces the cons operator with some other function. This takes some explaining.

- **foldr** in effect replaces the cons operator with another function of our choosing.
- This is similar to `Enum.reduce` in Elixir.
- The empty list is replaced with some initial value.

`foldr (+) 0 [1, 2, 3, 4, 5]`



Three arguments: **function**, **initial value**, **list**

foldl, foldr

Replaces the cons operator with some other function. This takes some explaining.

`foldr (+) 0 [1, 2, 3, 4, 5]`

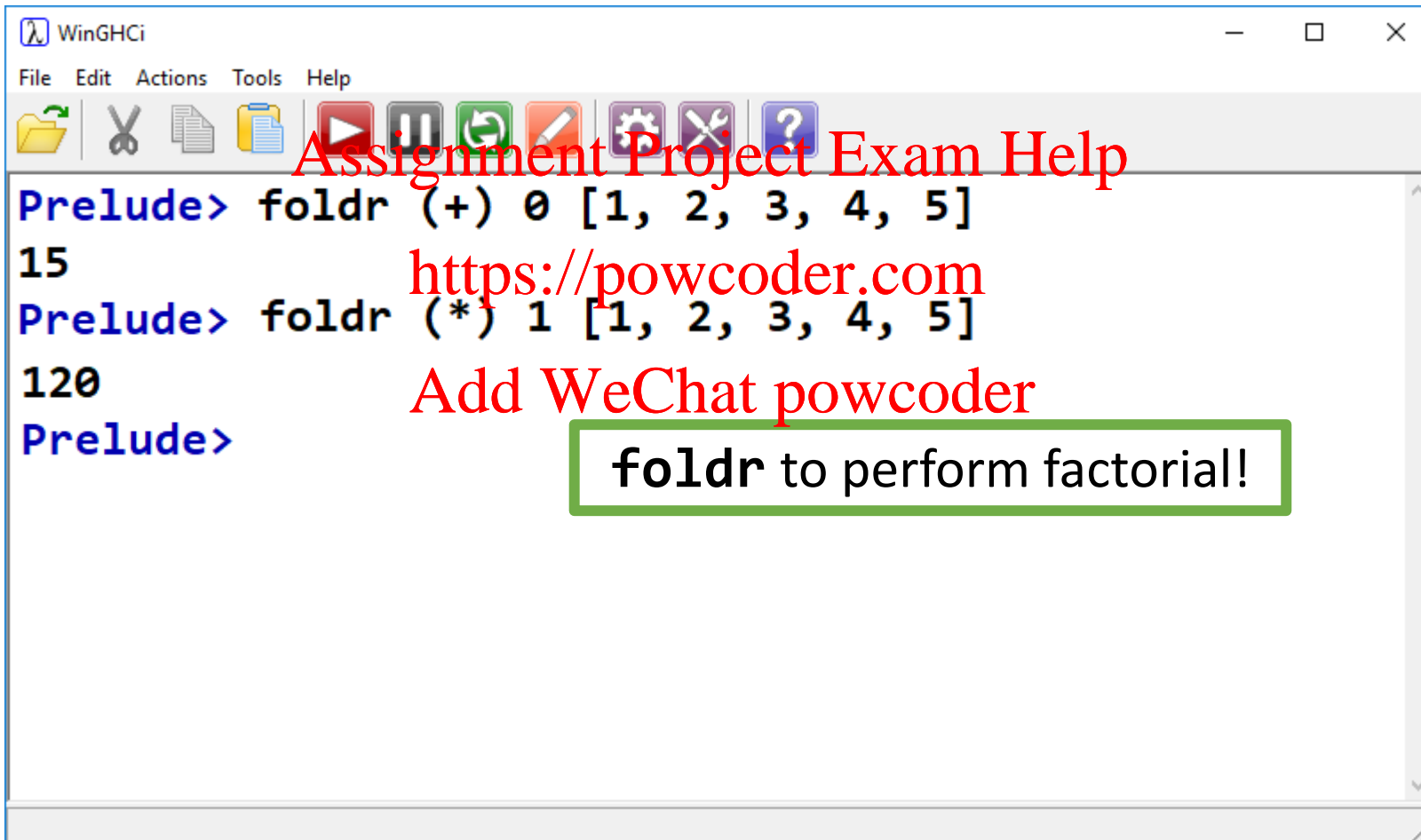
<https://powcoder.com>

`foldr (+) 0 1:2:3:4:5:[]`

`1 + 2 + 3 + 4 + 5 + 0`

`15`

foldl, foldr

A screenshot of a WinGHCi terminal window. The window has a title bar 'WinGHCi' and a menu bar 'File Edit Actions Tools Help'. Below the menu bar is a toolbar with icons for file operations and execution. The terminal shows the following commands and output:

```
Prelude> foldr (+) 0 [1, 2, 3, 4, 5]
15
Prelude> foldr (*) 1 [1, 2, 3, 4, 5]
120
Prelude>
```

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

foldr to perform factorial!

foldl VS foldr

foldr is *right associative*. Meaning:

`foldr (+) 0 [1, 2, 3, 4, 5]`

Assignment Project Exam Help

<https://powcoder.com>

1 + 2 + 3 + 4 + 5 + 0

Add WeChat powcoder

Is actually:

$(1 + (2 + (3 + (4 + (5 + 0)))))$

Doesn't matter for addition, but subtraction...

foldl VS foldr

foldr is *right associative*. Meaning:

`foldr (-) 1 [4, 8, 5]`

Assignment Project Exam Help

<https://powcoder.com>

4 - 8 - 5 - 1
Add WeChat powcoder

Is actually:

$(4 - (8 - (5 - 1)))$



0

foldl VS foldr

foldl is *left associative*. Meaning:

`foldl (-) 1 [4, 8, 5]`

Assignment Project Exam Help

<https://powcoder.com>

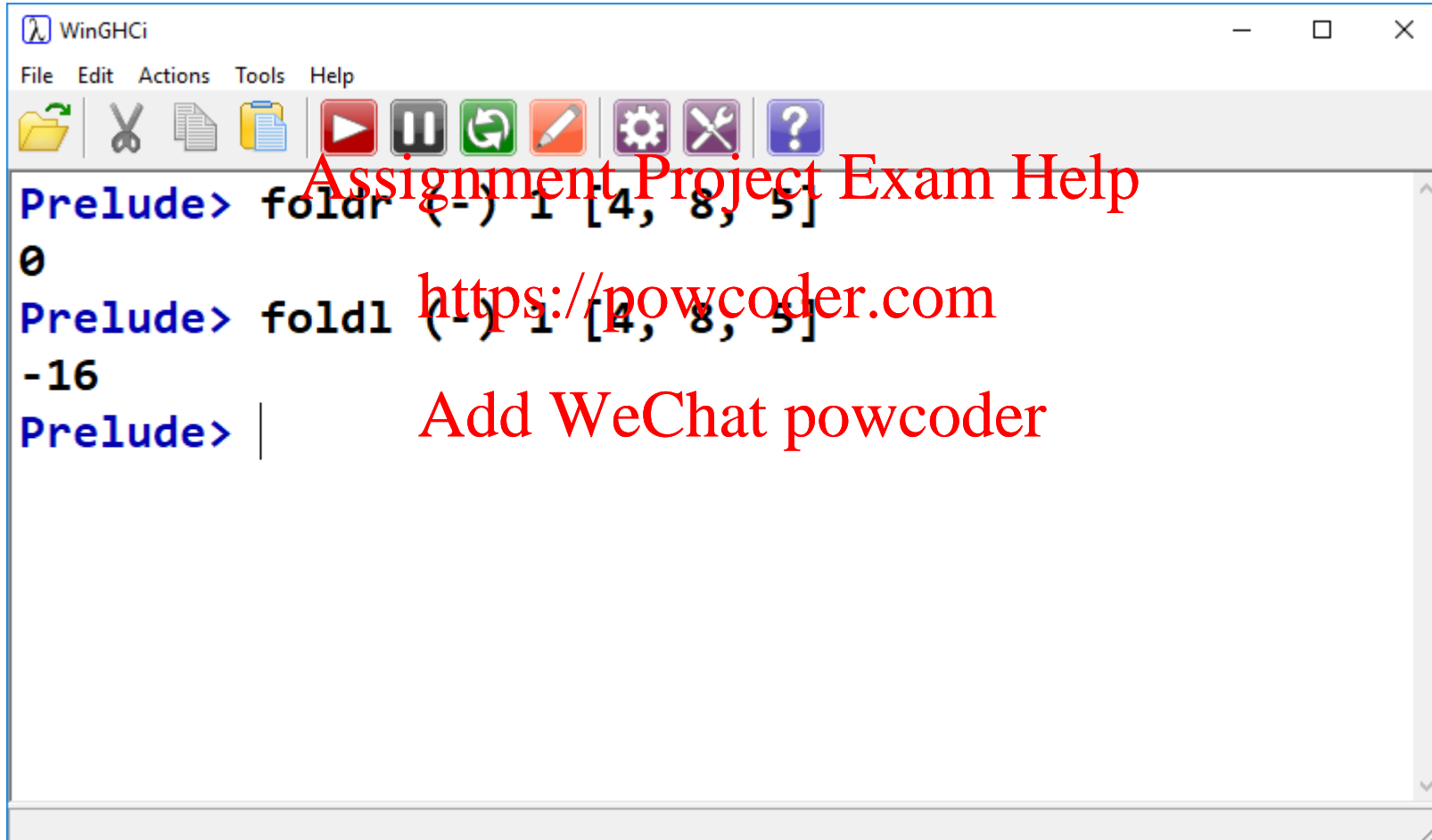
1 - 4 - 8 - 5
Add WeChat powcoder

Is actually:

$((1 - 4) - 8) - 5$

-16

foldl VS foldr

A screenshot of a WinGHCi terminal window. The window has a title bar 'WinGHCi' and a menu bar with 'File', 'Edit', 'Actions', 'Tools', and 'Help'. Below the menu bar is a toolbar with icons for file operations (folder, copy, paste), execution (play, pause, refresh), and settings (gear, wrench, question mark). The terminal text shows two commands: 'foldr (-) 1 [4, 8, 5]' which returns '0', and 'foldl (-) 1 [4, 8, 5]' which returns '-16'. The prompt 'Prelude>' is visible before each command and after the second result. A vertical scrollbar is on the right side of the terminal area.

```
Prelude> foldr (-) 1 [4, 8, 5]
0
Prelude> foldl (-) 1 [4, 8, 5]
-16
Prelude> |
```

List Generation

Syntactic sugar:

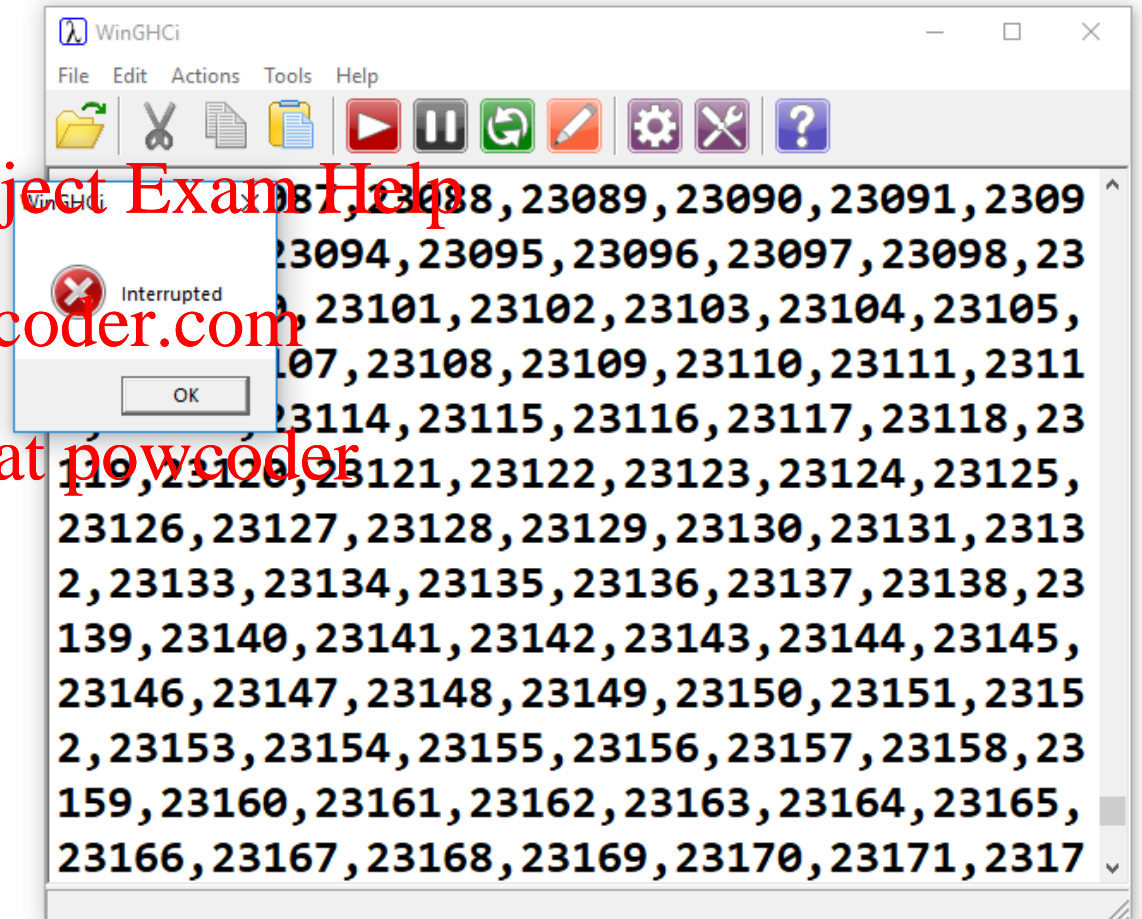
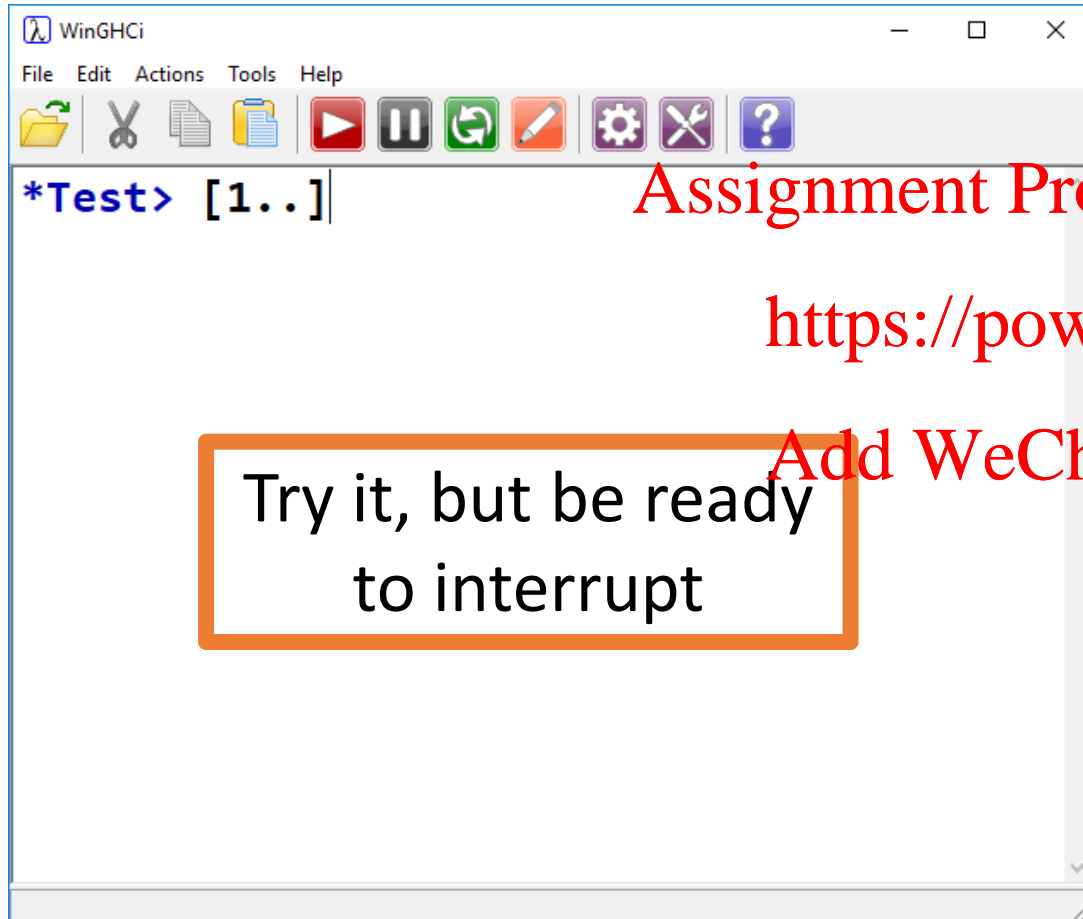
List declaration: `list = [1, 2, 3, 4, 5, 6, 7, 8, 9]`

Can be written: `list = [1, 9]`

Specify interval: `list = [1, 9]`
`= [1, 3, 5, 7, 9]`

Interval is discerned from difference between first two elements

Infinite Lists?



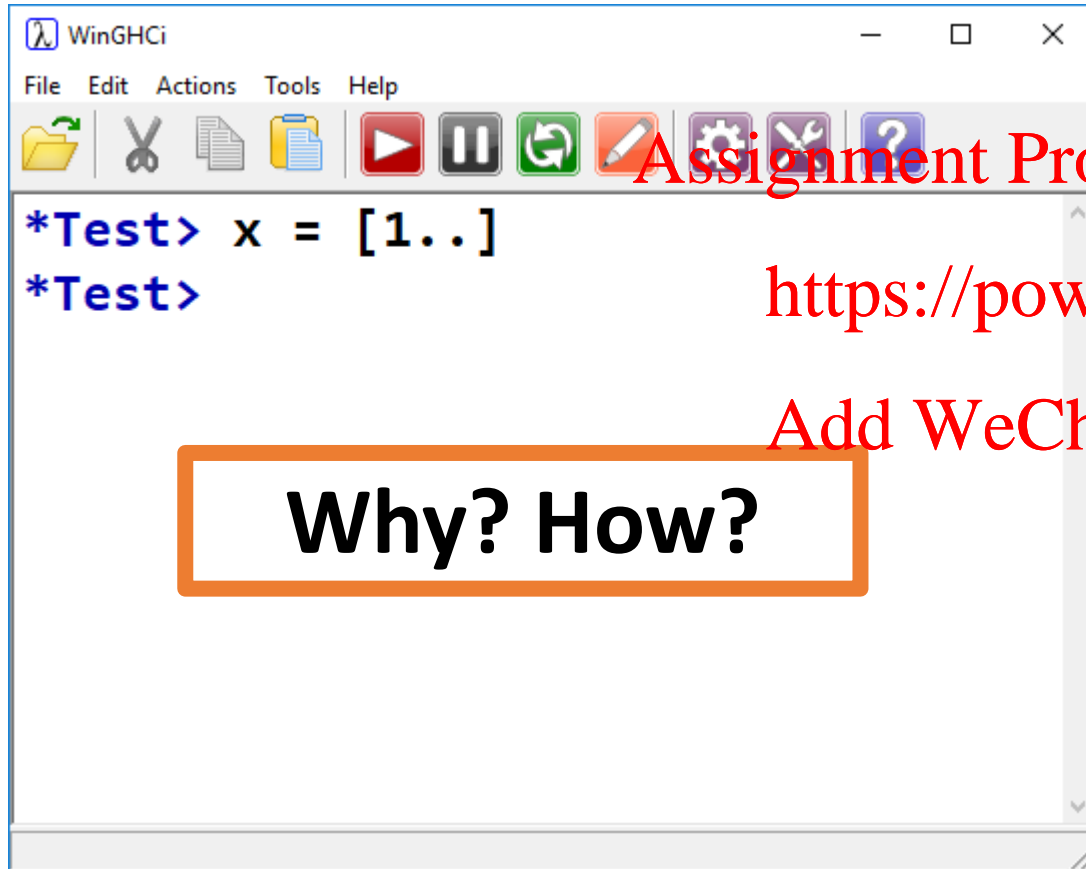
Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Try it, but be ready
to interrupt

Infinite Lists?



Assignment Project Exam Help

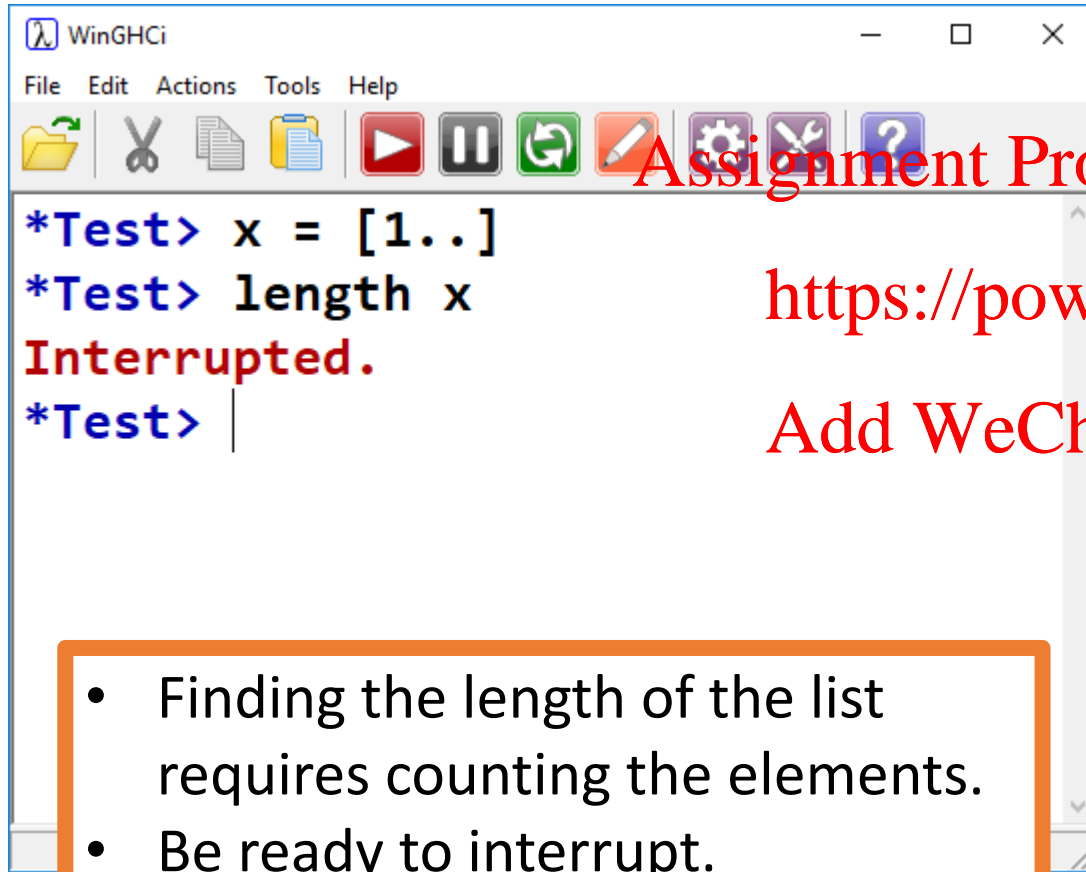
<https://powcoder.com>

Add WeChat powcoder

Haskell is lazy!

- We bind `x` to the expression to generate an infinite list.
- We don't have to *evaluate* this list to do so!
- Displaying the list, however, requires evaluation.

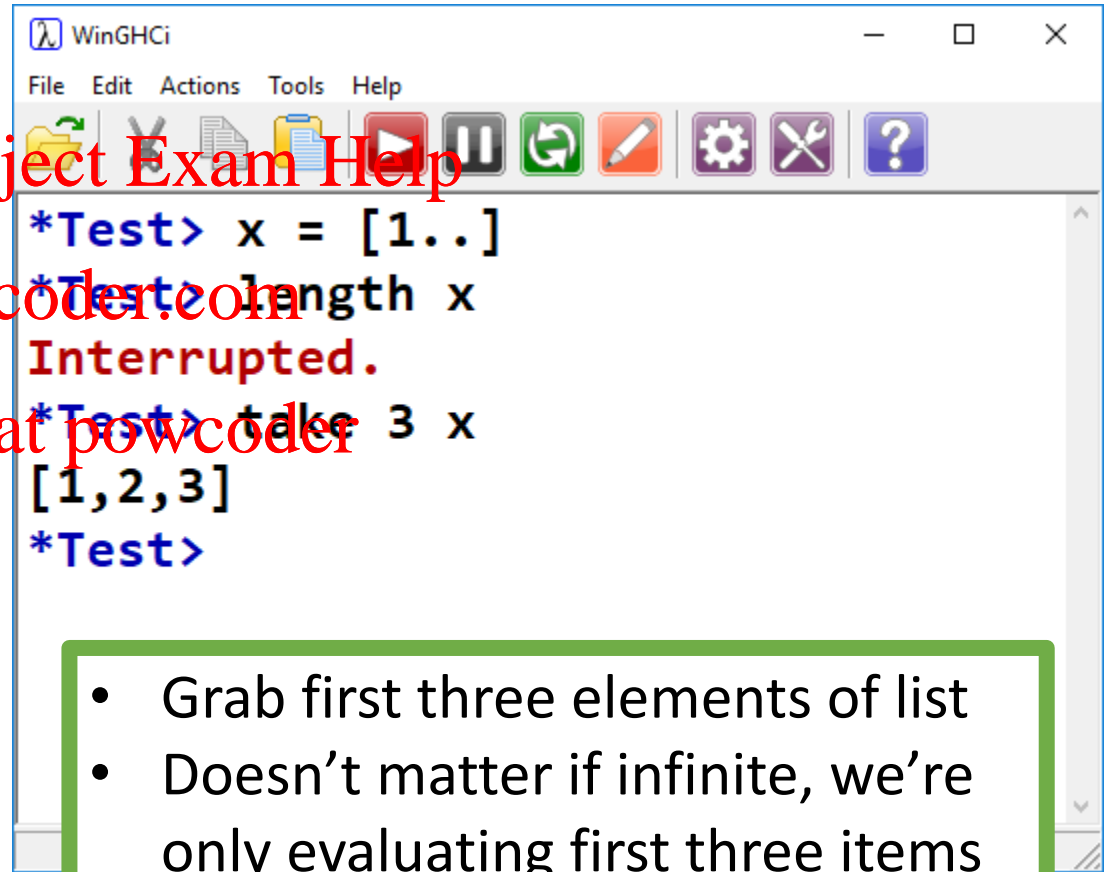
Infinite Lists?



```
WinGHCi
File Edit Actions Tools Help
[Icons]
*Test> x = [1..]
*Test> length x
Interrupted.
*Test> |
```

The terminal window shows the creation of an infinite list `x = [1..]`. When the command `length x` is entered, it results in an `Interrupted.` error. The prompt `*Test>` is followed by a vertical bar `|`.

- Finding the length of the list requires counting the elements.
- Be ready to interrupt.



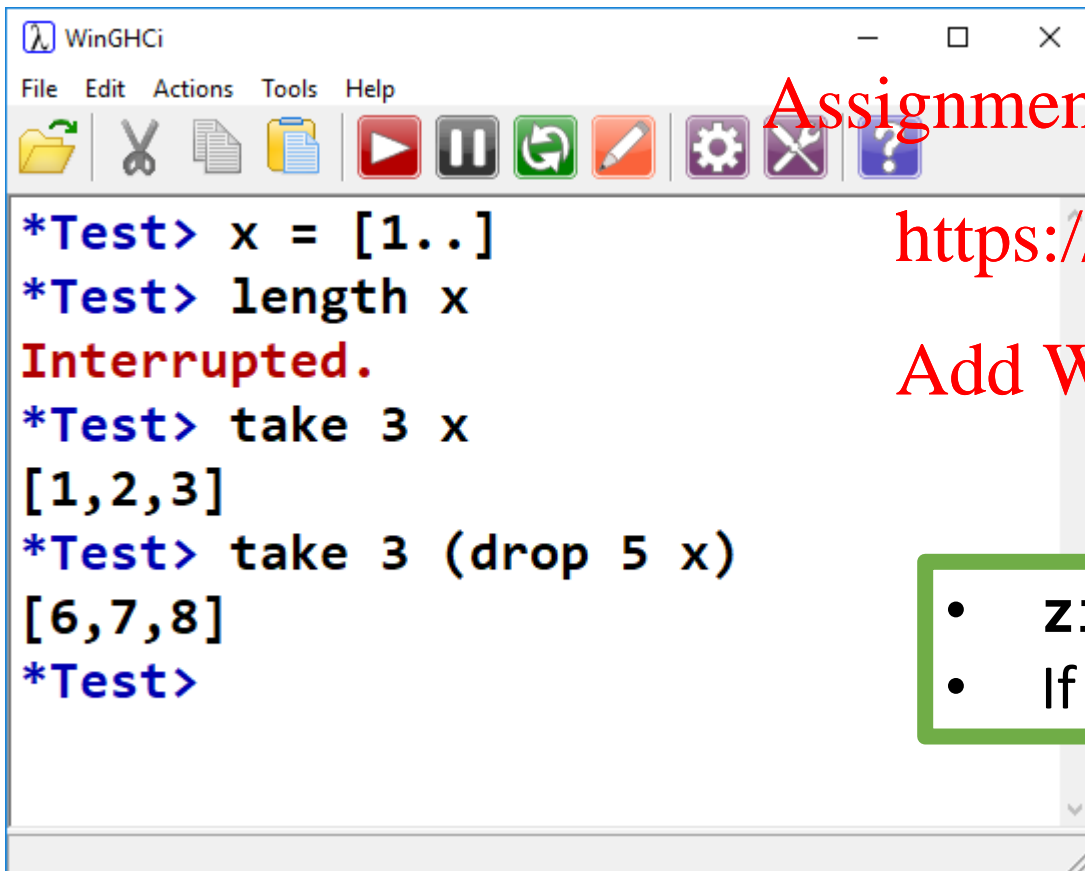
```
WinGHCi
File Edit Actions Tools Help
[Icons]
*Test> x = [1..]
*Test> length x
Interrupted.
*Test> take 3 x
[1,2,3]
*Test>
```

The terminal window shows the same infinite list `x = [1..]`. After an interrupted `length x` command, the `take 3 x` command is entered, which successfully returns the list `[1,2,3]`. The prompt `*Test>` is followed by a new line.

- Grab first three elements of list
- Doesn't matter if infinite, we're only evaluating first three items

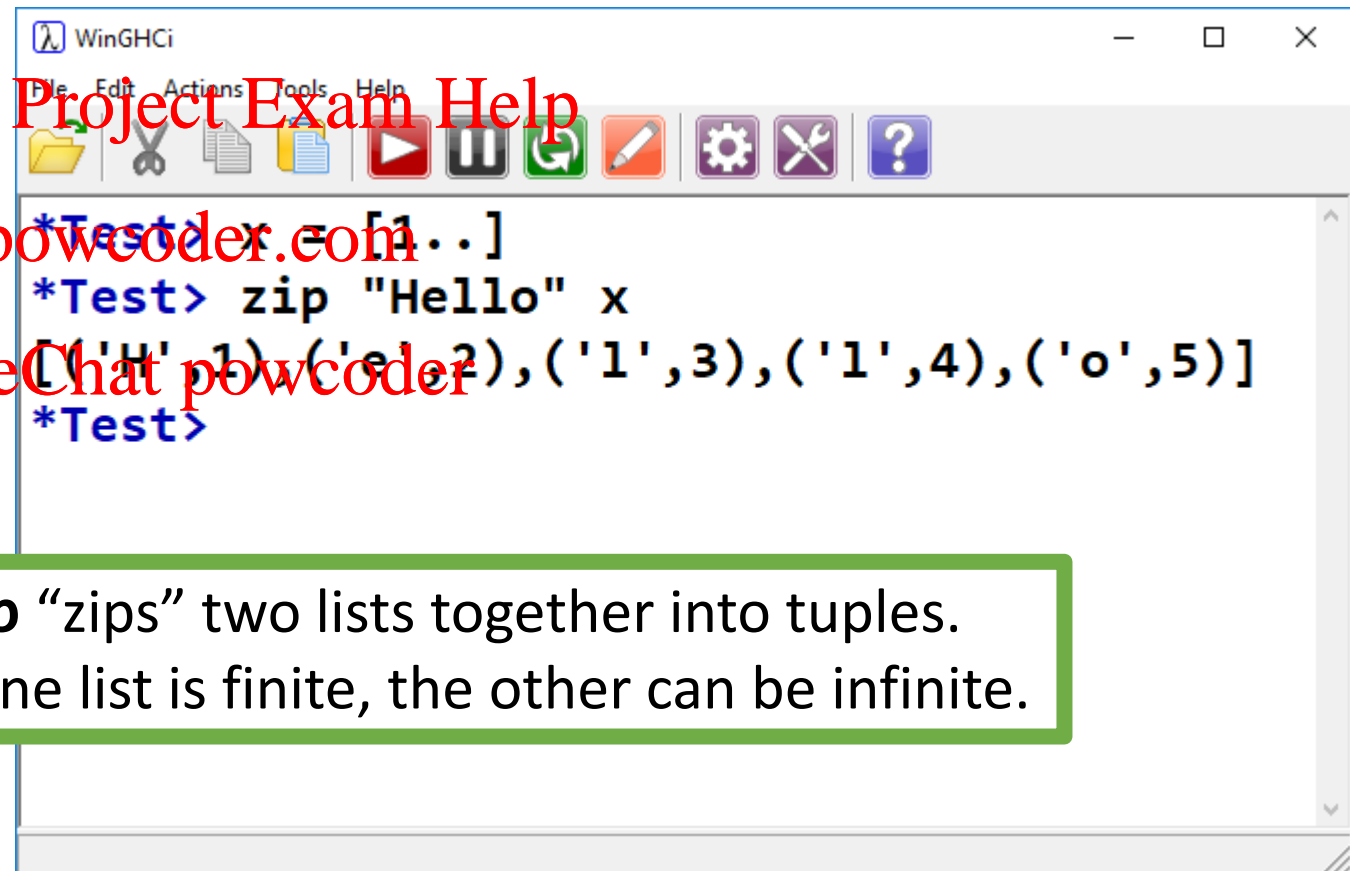
Infinite Lists?

We're allowed to perform operations on a *finite subset* of an infinite list.



```
WinGHCi
File Edit Actions Tools Help
[Icons]

*Test> x = [1..]
*Test> length x
Interrupted.
*Test> take 3 x
[1,2,3]
*Test> take 3 (drop 5 x)
[6,7,8]
*Test>
```



```
WinGHCi
File Edit Actions Tools Help
[Icons]

*Test> x = [1..]
*Test> zip "Hello" x
[('H',1),('e',2),('l',3),('l',4),('o',5)]
*Test>
```

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

- **zip** “zips” two lists together into tuples.
- If one list is finite, the other can be infinite.

Types in Haskell

Statically Typed:

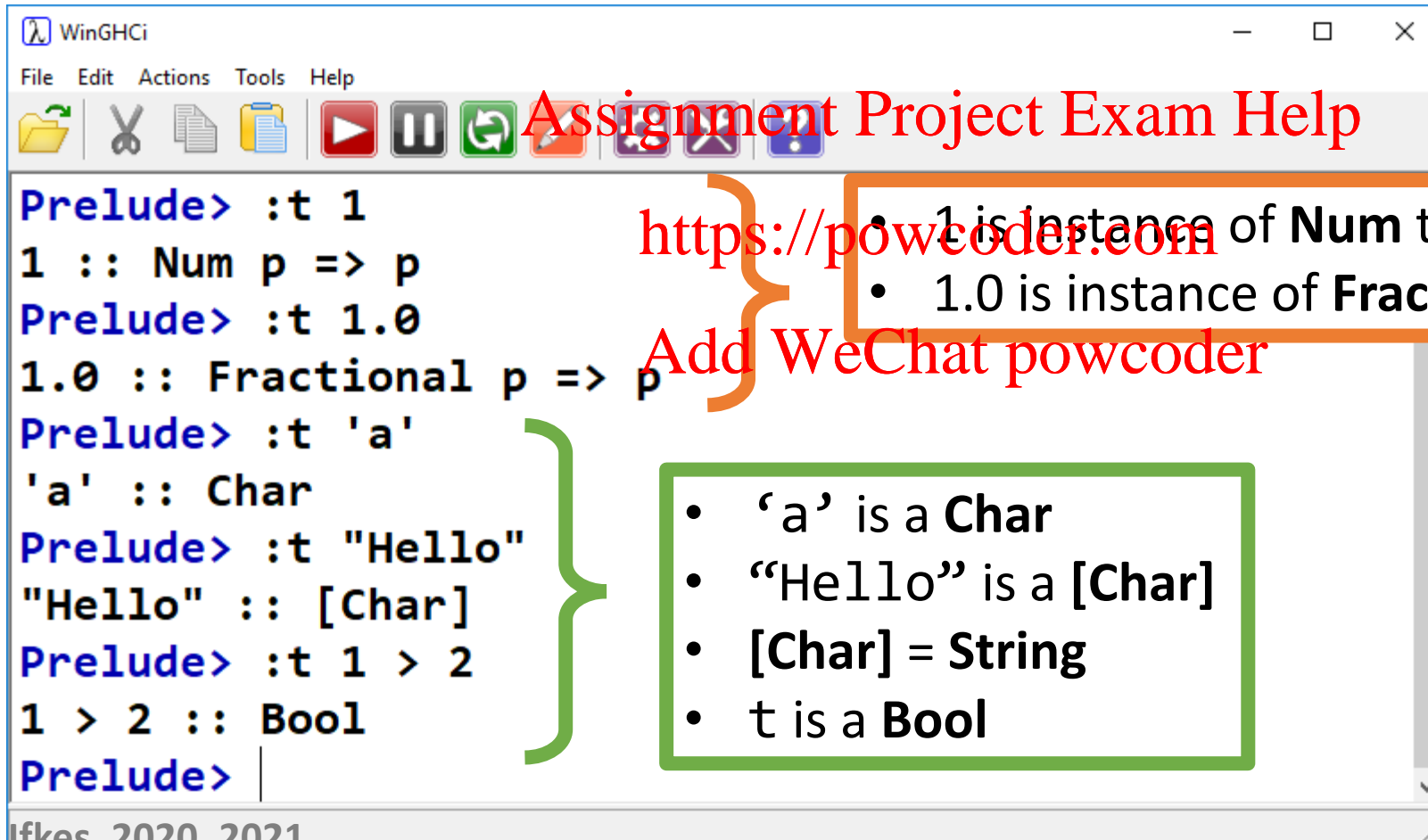
- Haskell uses static type checking.
- Every expression is assigned a type.
- If a function's arguments aren't the expected type, a compile error occurs.

Type Inference

- 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:



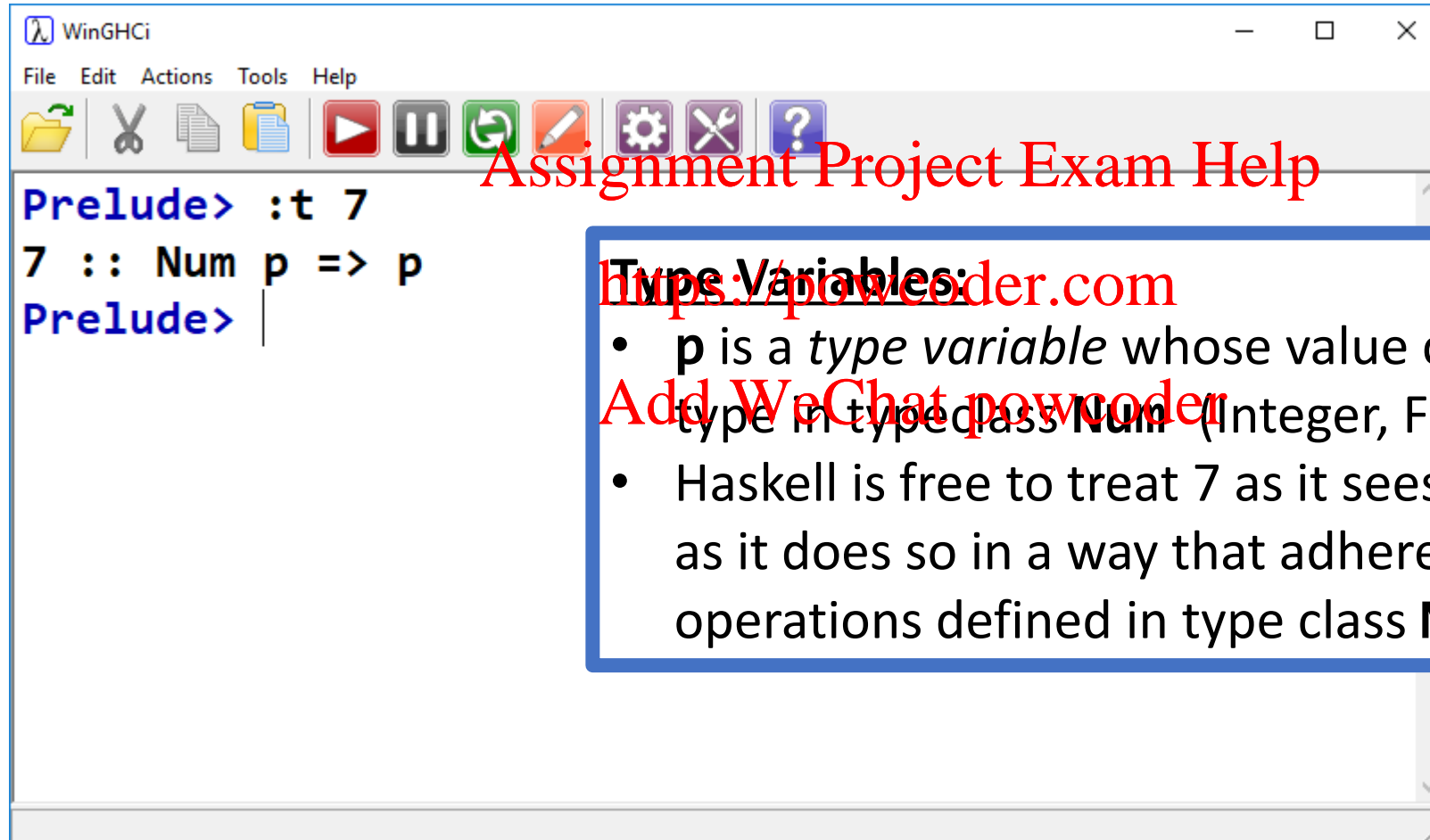
The screenshot shows the WinGHCi window with the following content:

```
WinGHCi
File Edit Actions Tools Help
[Icons]
Prelude> :t 1
1 :: Num p => p
Prelude> :t 1.0
1.0 :: Fractional p => p
Prelude> :t 'a'
'a' :: Char
Prelude> :t "Hello"
"Hello" :: [Char]
Prelude> :t 1 > 2
1 > 2 :: Bool
Prelude>
```

Annotations on the image:

- A red watermark "Assignment Project Exam Help" is overlaid on the top right.
- A red bracket groups the first two queries, with a link <https://powcoder.com> and the text "Add WeChat powcoder" next to it.
- An orange box highlights the text: "1 is instance of **Num** type class." and "1.0 is instance of **Fractional** type class."
- A green bracket groups the last three queries, with a green box containing the following list:
 - 'a' is a **Char**
 - "Hello" is a **[Char]**
 - **[Char]** = **String**
 - `t` is a **Bool**

Num p => p ?



A screenshot of the WinGHCi Haskell interpreter window. The window has a menu bar with 'File', 'Edit', 'Actions', 'Tools', and 'Help'. Below the menu is a toolbar with icons for file operations, execution, and help. The main text area shows the following interaction:

```
Prelude> :t 7
7 :: Num p => p
Prelude> |
```

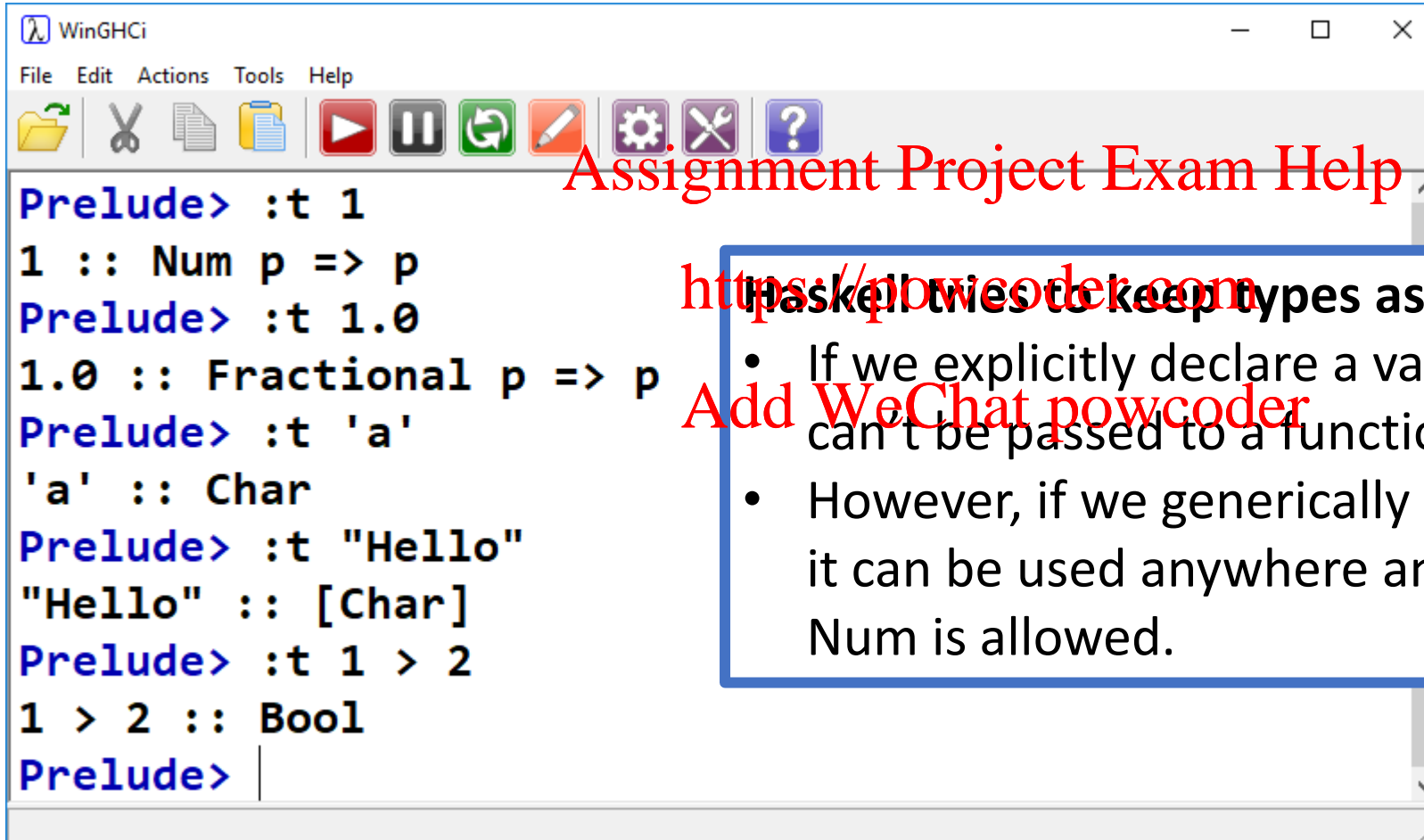
Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

- **p** is a *type variable* whose value can be any type in typeclass **Num** (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?



```
WinGHCi
File Edit Actions Tools Help
[Paste] [Cut] [Copy] [Run] [Pause] [Refresh] [Undo] [Redo] [Settings] [Exit] [Help]

Prelude> :t 1
1 :: Num p => p
Prelude> :t 1.0
1.0 :: Fractional p => p
Prelude> :t 'a'
'a' :: Char
Prelude> :t "Hello"
"Hello" :: [Char]
Prelude> :t 1 > 2
1 > 2 :: Bool
Prelude> 
```

Assignment Project Exam Help

<https://powcoder.com>

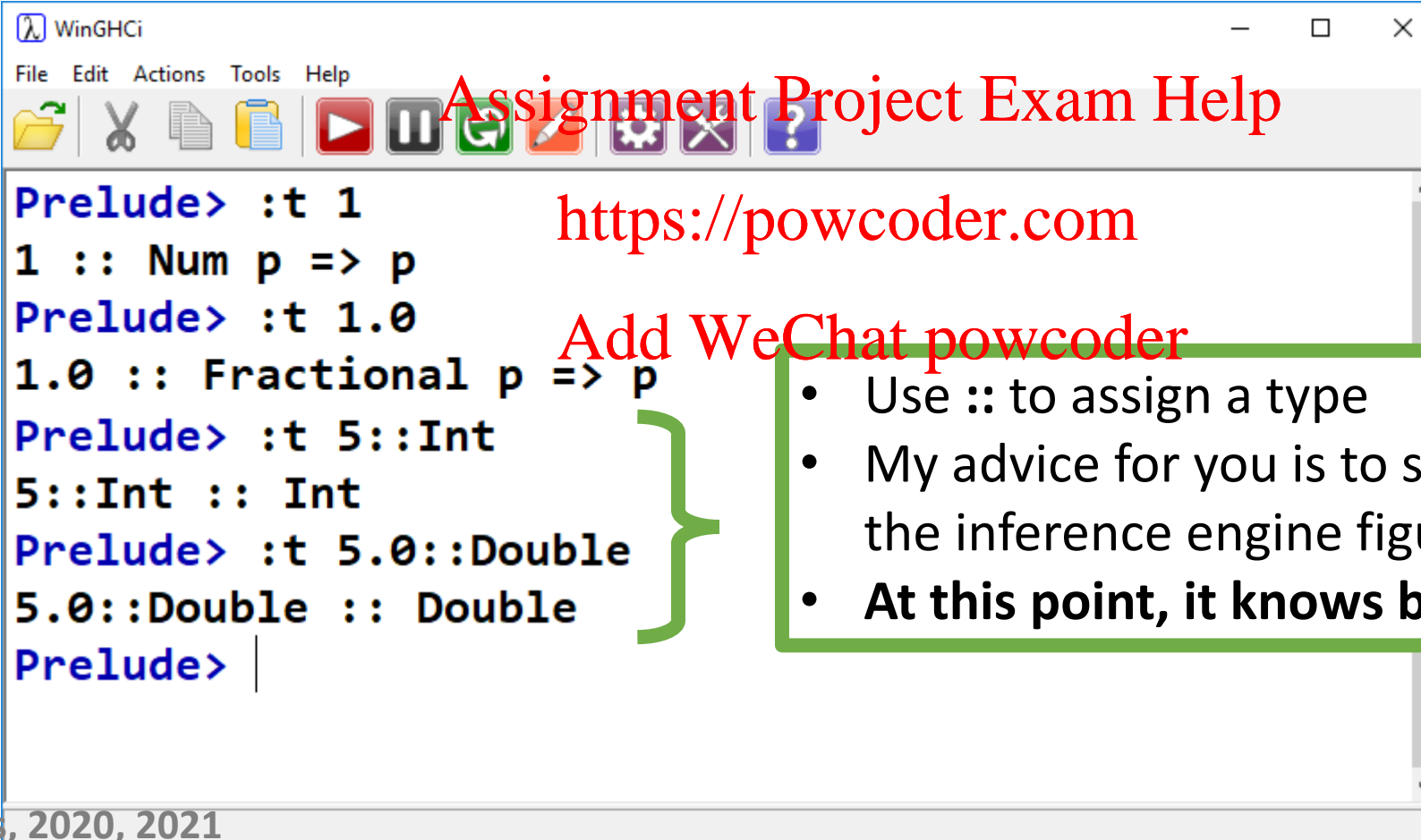
Add WeChat powcoder

Haskell tries to keep types as generic as possible

- If we explicitly declare a variable as integer, it can't be passed to a function requiring 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:



The screenshot shows the WinGHCi window with the following content:

```
WinGHCi
File Edit Actions Tools Help
Prelude> :t 1
1 :: Num p => p
Prelude> :t 1.0
1.0 :: Fractional p => p
Prelude> :t 5::Int
5::Int :: Int
Prelude> :t 5.0::Double
5.0::Double :: Double
Prelude> |
```

Red text overlays on the image include:

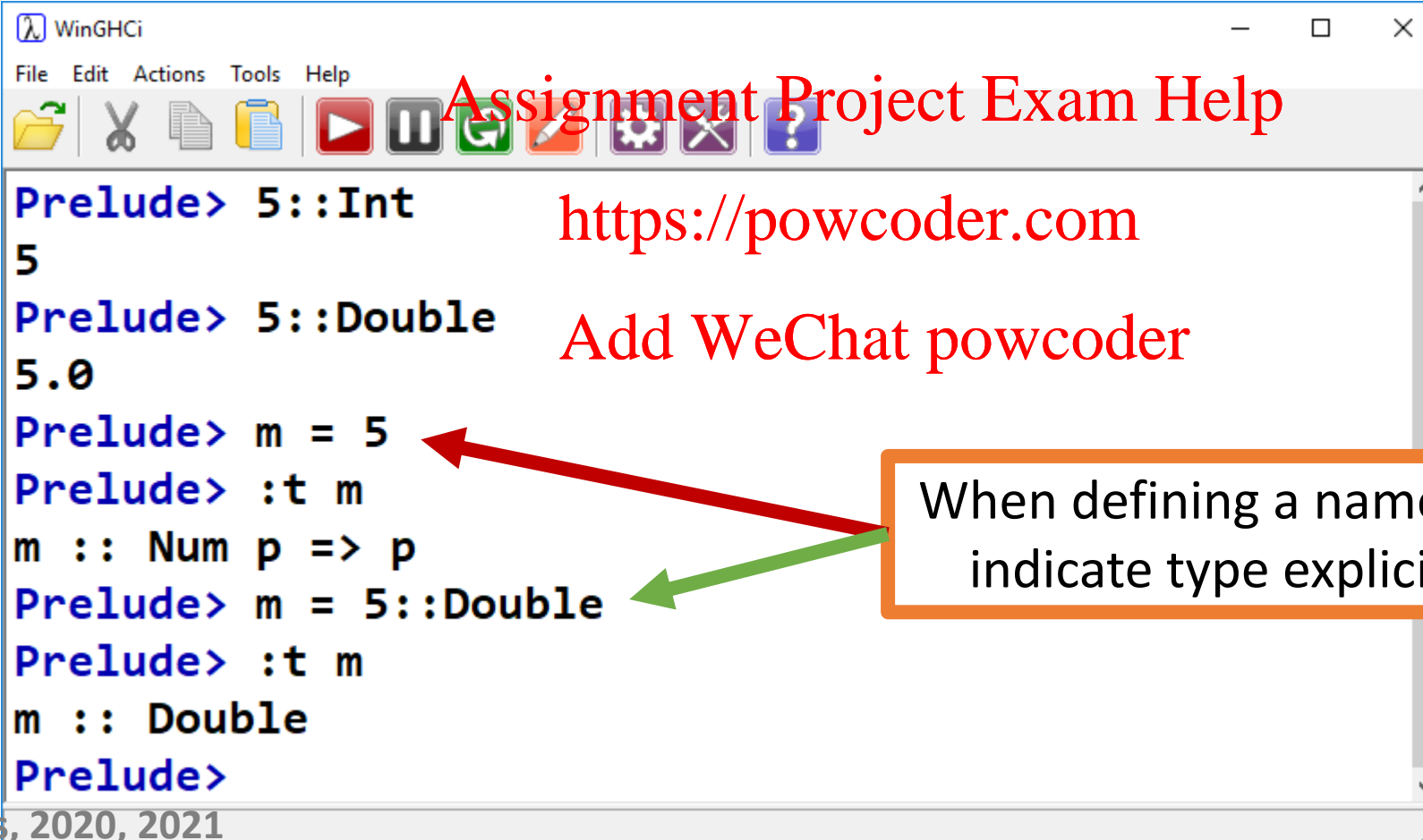
- Assignment Project Exam Help
- <https://powcoder.com>
- Add WeChat powcoder

A green box on the right contains the following list:

- Use `::` to assign 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:



The screenshot shows the WinGHCi window with the following interactions:

```
Prelude> 5::Int
5
Prelude> 5::Double
5.0
Prelude> m = 5
Prelude> :t m
m :: Num p => p
Prelude> m = 5::Double
Prelude> :t m
m :: Double
Prelude>
```

Annotations and links:

- Red text: <https://powcoder.com>
- Red text: Add WeChat powcoder
- Orange box: When defining a name, can indicate type explicitly:

Arrows point from the orange box to the lines `m = 5` and `m = 5::Double`.

Type Classes

Type polymorphism and type variables:

Recall: Overloading ~~Assignment Project Exam Help~~

- In languages like C++, the `==` operator is overloaded to work with many different types.
- Numeric type equality and string equality are performed differently.
- In general, if we want to compare two values of type α , we use an **α -compare**
- α 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 α), and returns a Boolean

<https://powcoder.com>
This operator may not be defined for *all* types, just some.

Add WeChat powcoder

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

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

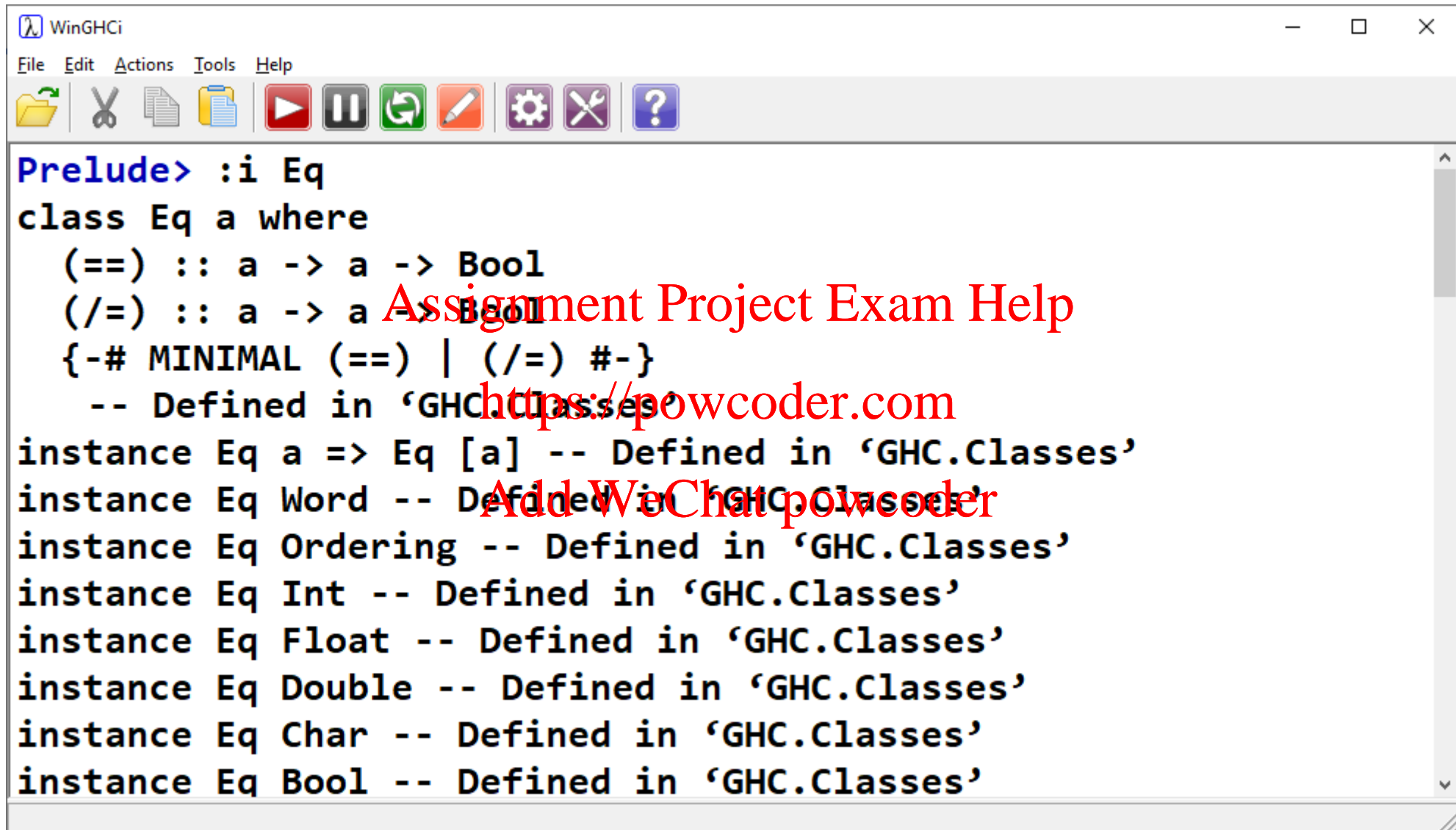
Eq Type Class

`(==)` is defined for types
in typeclass **Eq**

`(==) :: Eq a => a -> a -> Bool`

- `(==)` takes two args of type **a**, where **a** is a member of type class **Eq**
- It returns **Bool**

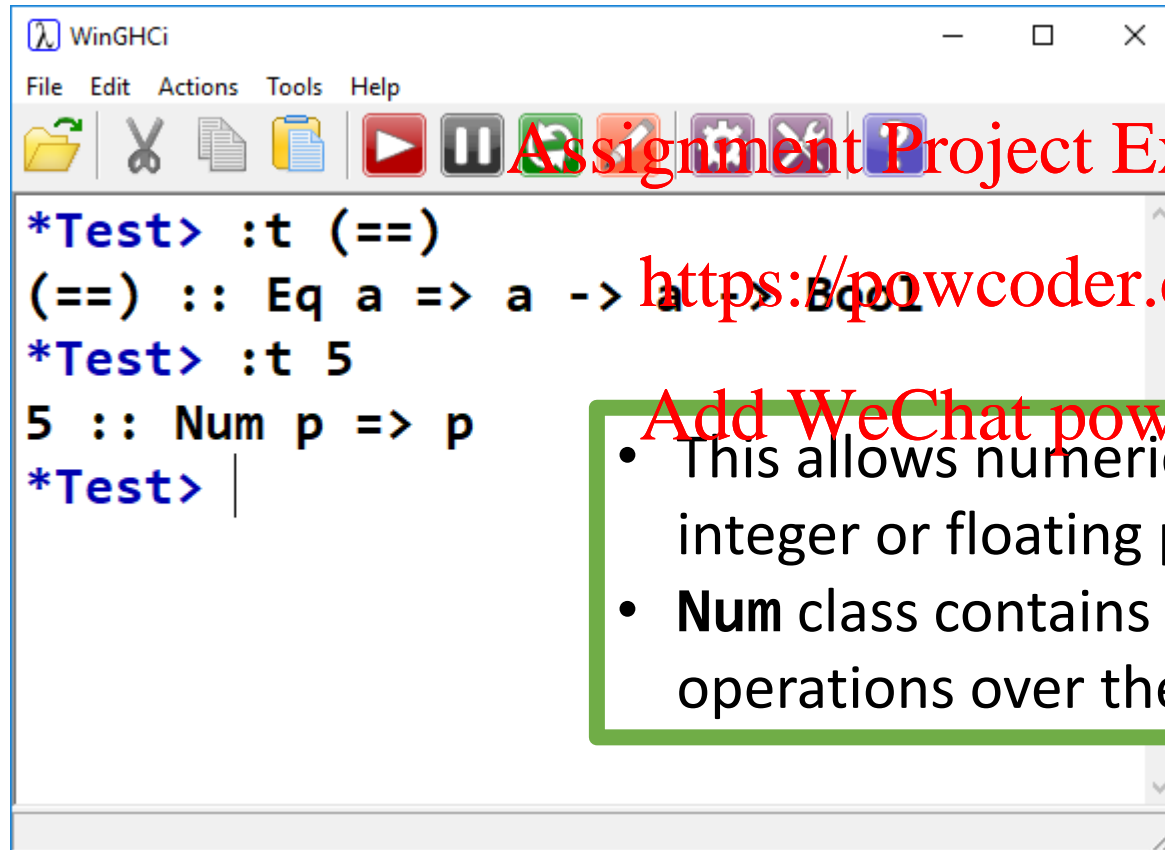
- If a concrete type, **a**, belongs to a certain type class, we say **a** is an *instance* of that type class.
- **Int** is an instance of **Eq**, for example.



The image shows a screenshot of the WinGHCi window. The title bar says 'WinGHCi'. The menu bar includes 'File', 'Edit', 'Actions', 'Tools', and 'Help'. The toolbar contains icons for file operations (copy, paste, save, etc.) and execution (run, pause, etc.). The main text area displays the following Haskell code:

```
Prelude> :i Eq
class Eq a where
  (==) :: a -> a -> Bool
  (/=) :: a -> a -> Bool
  {-# MINIMAL (==) | (/=) #-}
  -- Defined in 'GHC.Classes'
instance Eq a => Eq [a] -- Defined in 'GHC.Classes'
instance Eq Word -- Defined in 'GHC.Classes'
instance Eq Ordering -- Defined in 'GHC.Classes'
instance Eq Int -- Defined in 'GHC.Classes'
instance Eq Float -- Defined in 'GHC.Classes'
instance Eq Double -- Defined in 'GHC.Classes'
instance Eq Char -- Defined in 'GHC.Classes'
instance Eq Bool -- Defined in 'GHC.Classes'
```

Num Type Class



```
WinGHCi
File Edit Actions Tools Help
*Test> :t (==)
(==) :: Eq a => a -> a -> Bool
*Test> :t 5
5 :: Num p => p
*Test> |
```

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

- This allows numeric values 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

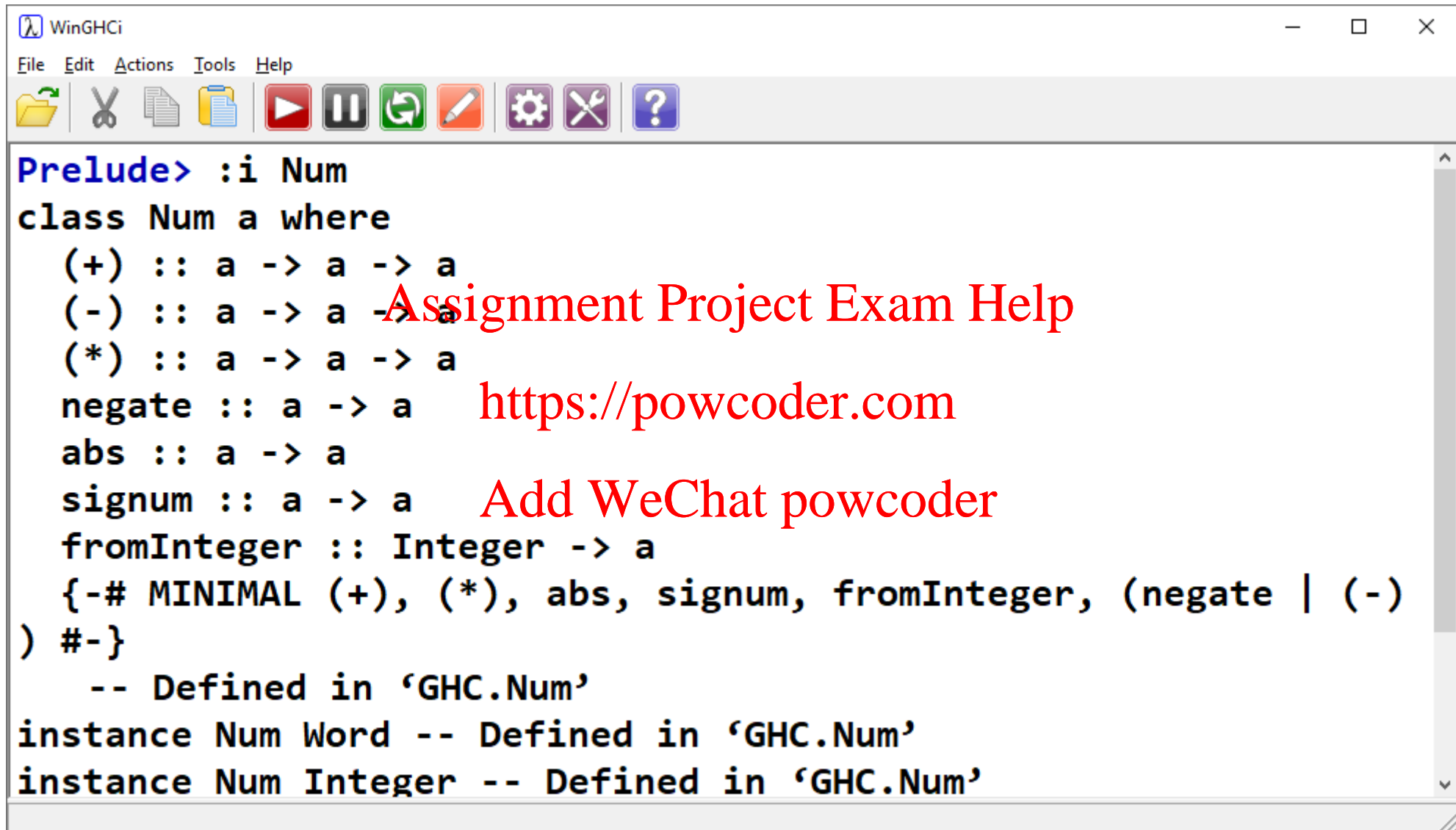
The screenshot shows the WinGHCi window with the following text:

```
*Test> :t (==)
(==) :: Eq a => a -> a -> Bool
*Test> :t 5
5 :: Num p => p
*Test>
```

Red text overlays the image:

- Assignment Project Exam Help
- <https://powcoder.com>
- Add WeChat powcoder

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

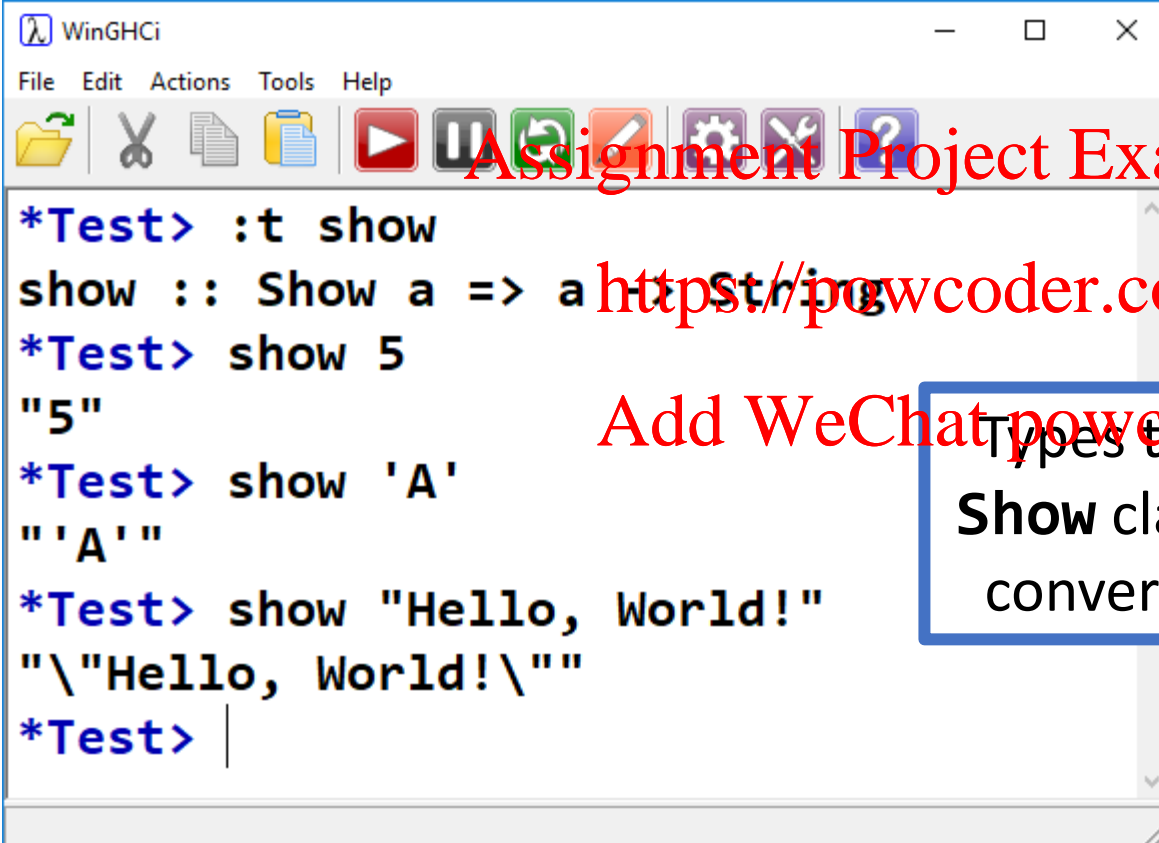


The image shows a screenshot of a WinGHCi window. The window has a title bar with the WinGHCi logo and standard window controls. Below the title bar is a menu bar with 'File', 'Edit', 'Actions', 'Tools', and 'Help'. Under the 'Actions' menu, there is a toolbar with icons for file operations (copy, paste, save), execution (run, step through, step over), and help (question mark). The main text area contains Haskell code defining a `Num` class. The code includes type signatures for `(+)`, `(-)`, `(*)`, `negate`, `abs`, and `signum`, as well as `fromInteger`. It also includes a `MINIMAL` pragma and two `instance` declarations for `Word` and `Integer`. Overlaid on the right side of the code is red text providing a website and a WeChat contact.

```
Prelude> :i Num
class Num a where
  (+) :: a -> a -> a
  (-) :: a -> a -> a
  (*) :: a -> a -> a
  negate :: a -> a
  abs :: a -> a
  signum :: a -> a
  fromInteger :: Integer -> a
  {-# MINIMAL (+), (*), abs, signum, fromInteger, (negate | (-)
) #-}
  -- Defined in 'GHC.Num'
instance Num Word -- Defined in 'GHC.Num'
instance Num Integer -- Defined in 'GHC.Num'
```

Assignment Project Exam Help
<https://powcoder.com>
Add WeChat powcoder

Show Type Class



A screenshot of the WinGHCi terminal window. The window has a title bar 'WinGHCi' and a menu bar with 'File', 'Edit', 'Actions', 'Tools', and 'Help'. Below the menu bar is a toolbar with icons for file operations and execution. The terminal shows the following interactions:

```
*Test> :t show
show :: Show a => a -> String
*Test> show 5
"5"
*Test> show 'A'
"'A'"
*Test> show "Hello, World!"
 "\"Hello, World!\""
*Test> |
```

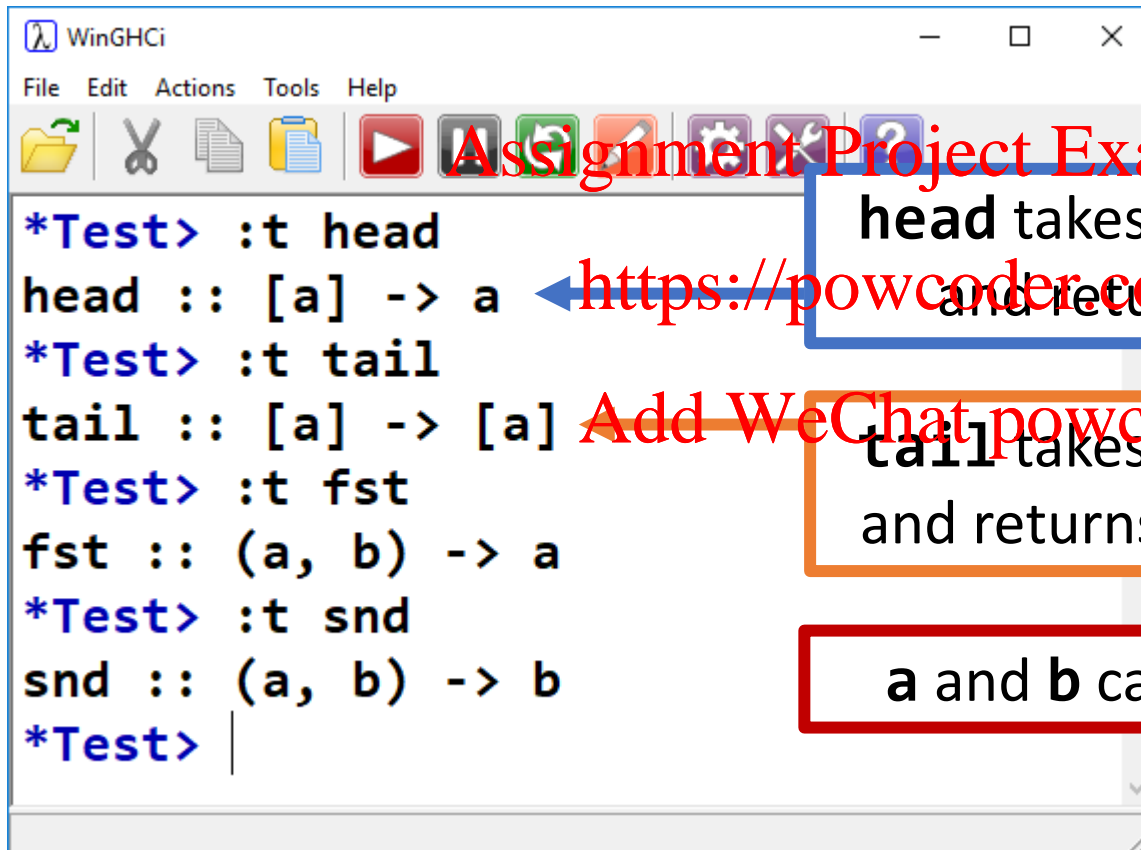
Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Types that are members of the **Show** class have functions which convert their value to a String.

Function Types



```
WinGHCi
File Edit Actions Tools Help
*Test> :t head
head :: [a] -> a
*Test> :t tail
tail :: [a] -> [a]
*Test> :t fst
fst :: (a, b) -> a
*Test> :t snd
snd :: (a, b) -> b
*Test> |
```

Assignment Project Exam Help

<https://powcoder.com>

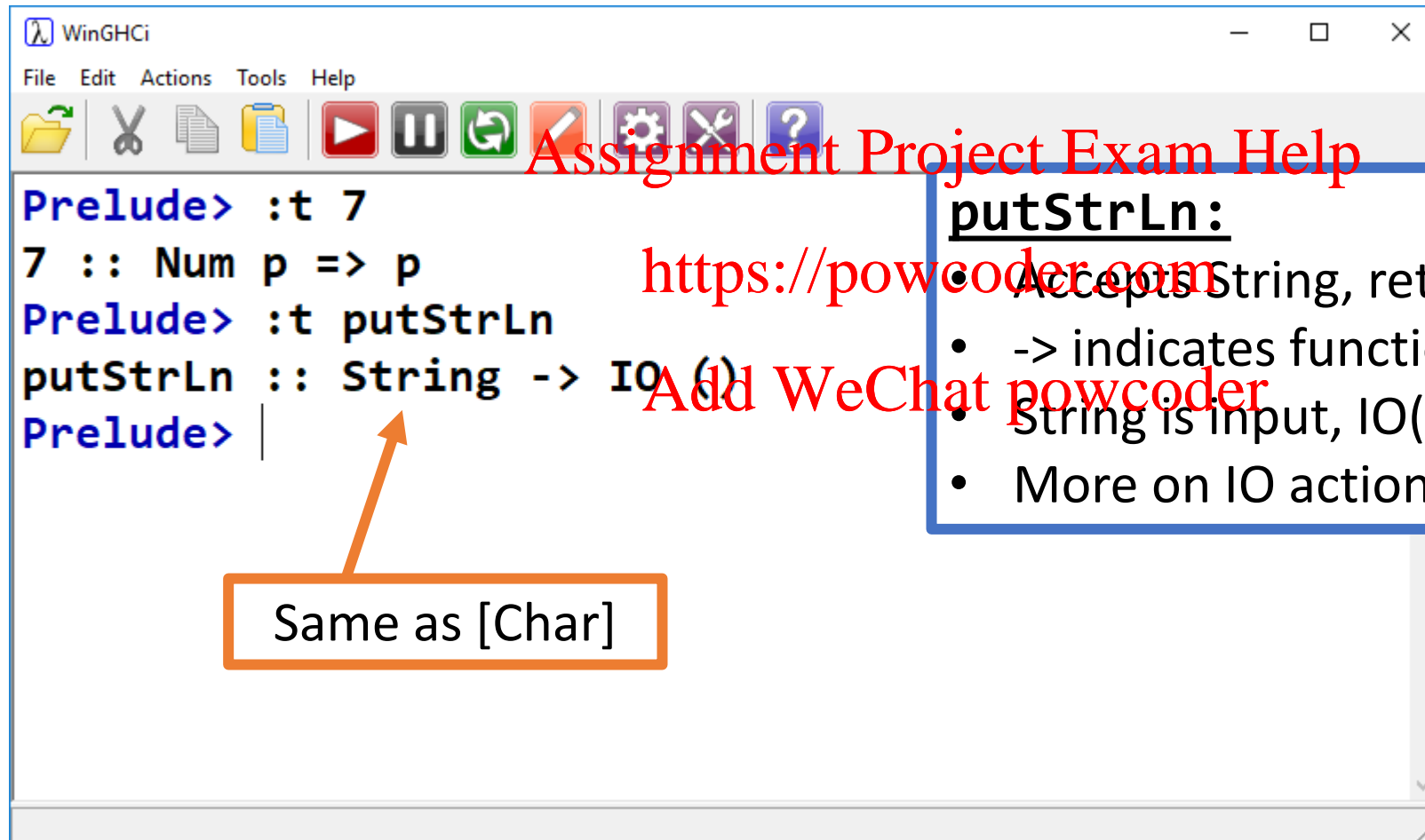
Add WeChat powcoder

head takes a list containing type **a**,
and returns a value of type **a**

tail takes a list containing type **a**,
and returns a list containing type **a**

a and **b** can be *literally any type*!

Function Types



```
WinGHCi
File Edit Actions Tools Help
[Icons]

Prelude> :t 7
7 :: Num p => p
Prelude> :t putStrLn
putStrLn :: String -> IO ()
Prelude> |
```

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

putStrLn:

- Accepts String, returns ***IO action***.
- -> indicates function
- String is input, IO() is output.
- More on IO actions later.

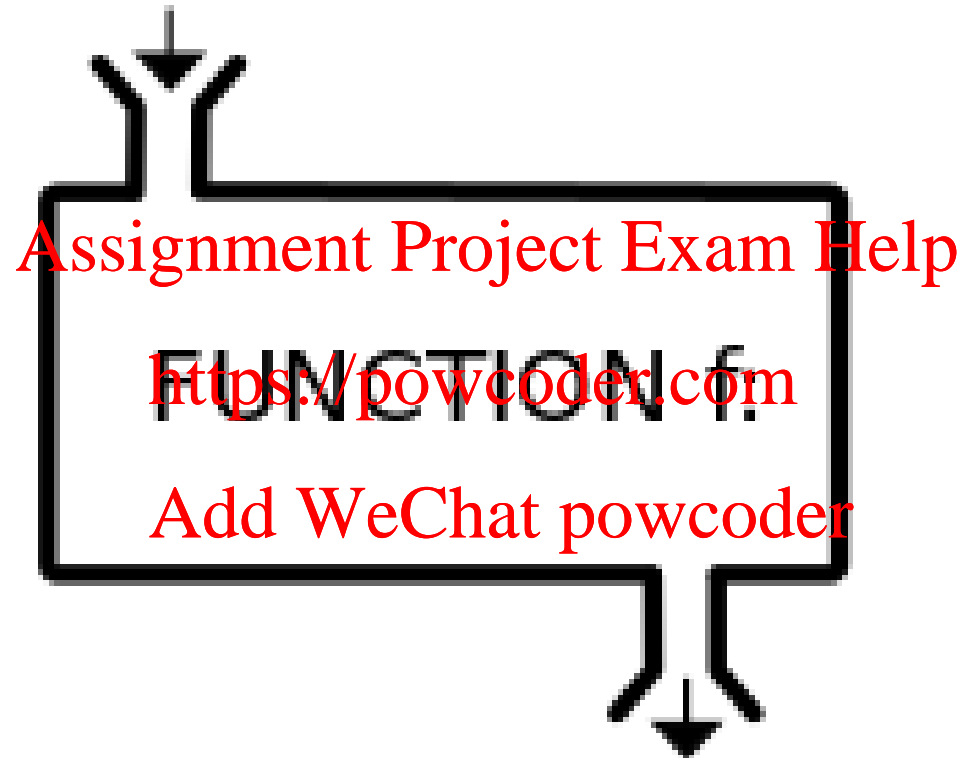
Same as [Char]

We'll create our own types soon, and see how to
add them to existing type classes.

<https://powcoder.com>

Add WeChat powcoder

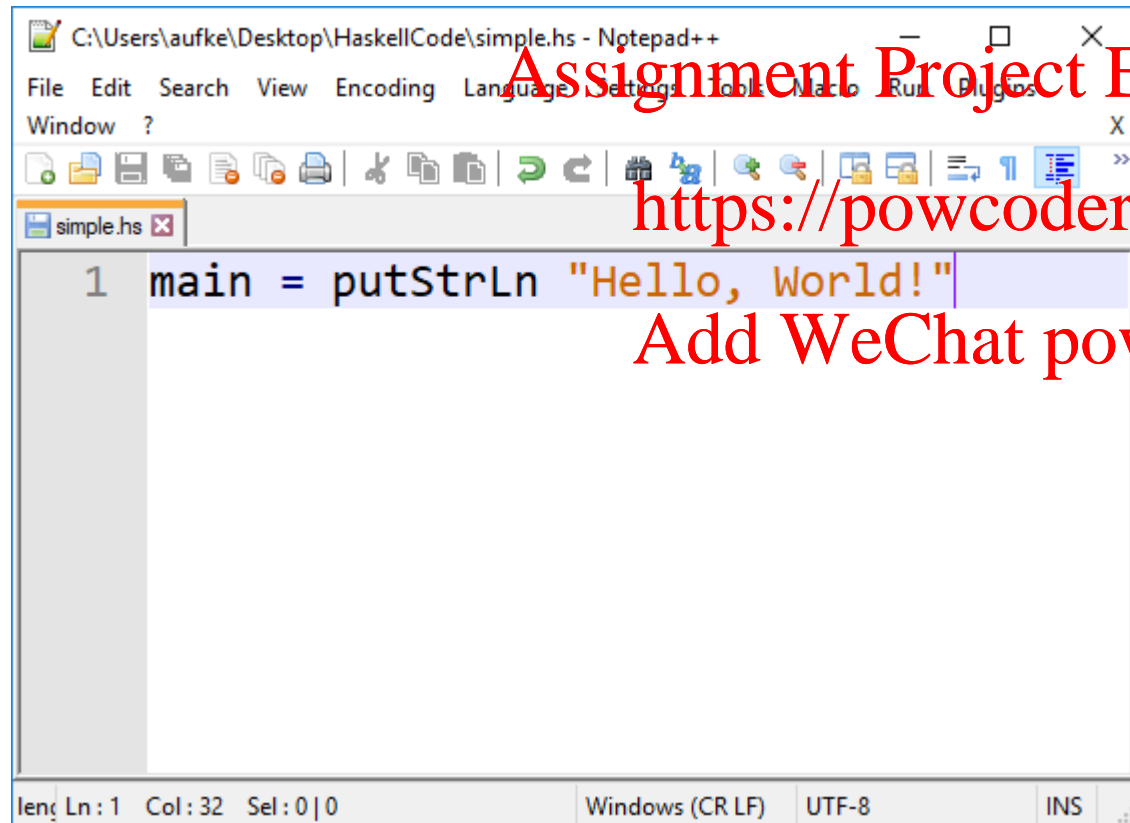
INPUT x



OUTPUT $f(x)$

Functions in Haskell

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



```
1 main = putStrLn "Hello, World!"
```

Assignment Project Exam Help

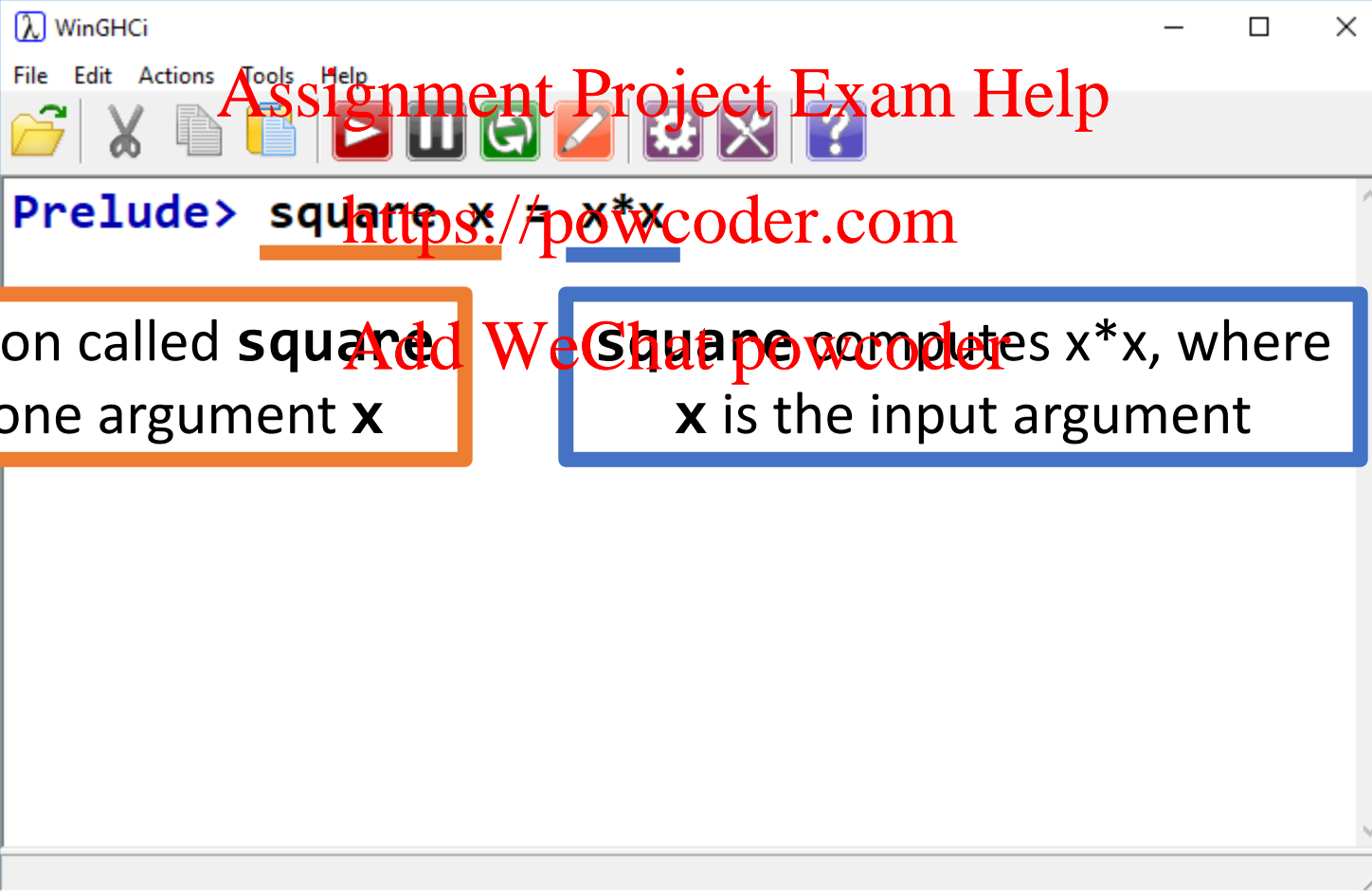
<https://powcoder.com>

Add WeChat powcoder

- If we're compiling our code into an executable, we need a main.
- If we're using the GHCi shell, we don't.

Functions in Haskell

Let's start simple:



The image shows a screenshot of the WinGHCi Haskell interpreter window. The window has a menu bar with 'File', 'Edit', 'Actions', 'Tools', and 'Help'. Below the menu bar is a toolbar with various icons. The main text area contains the code `Prelude> square x = x*x`. The word `square` is underlined in orange, and `x` is underlined in blue. Two callout boxes are present: an orange box on the left and a blue box on the right. The orange box contains the text 'Define function called `square` that takes one argument `x`'. The blue box contains the text 'Square computes $x*x$, where `x` is the input argument'. A large red watermark 'Assignment Project Exam Help' is overlaid across the top half of the window, and a smaller red watermark 'https://powcoder.com' is overlaid on the code line.

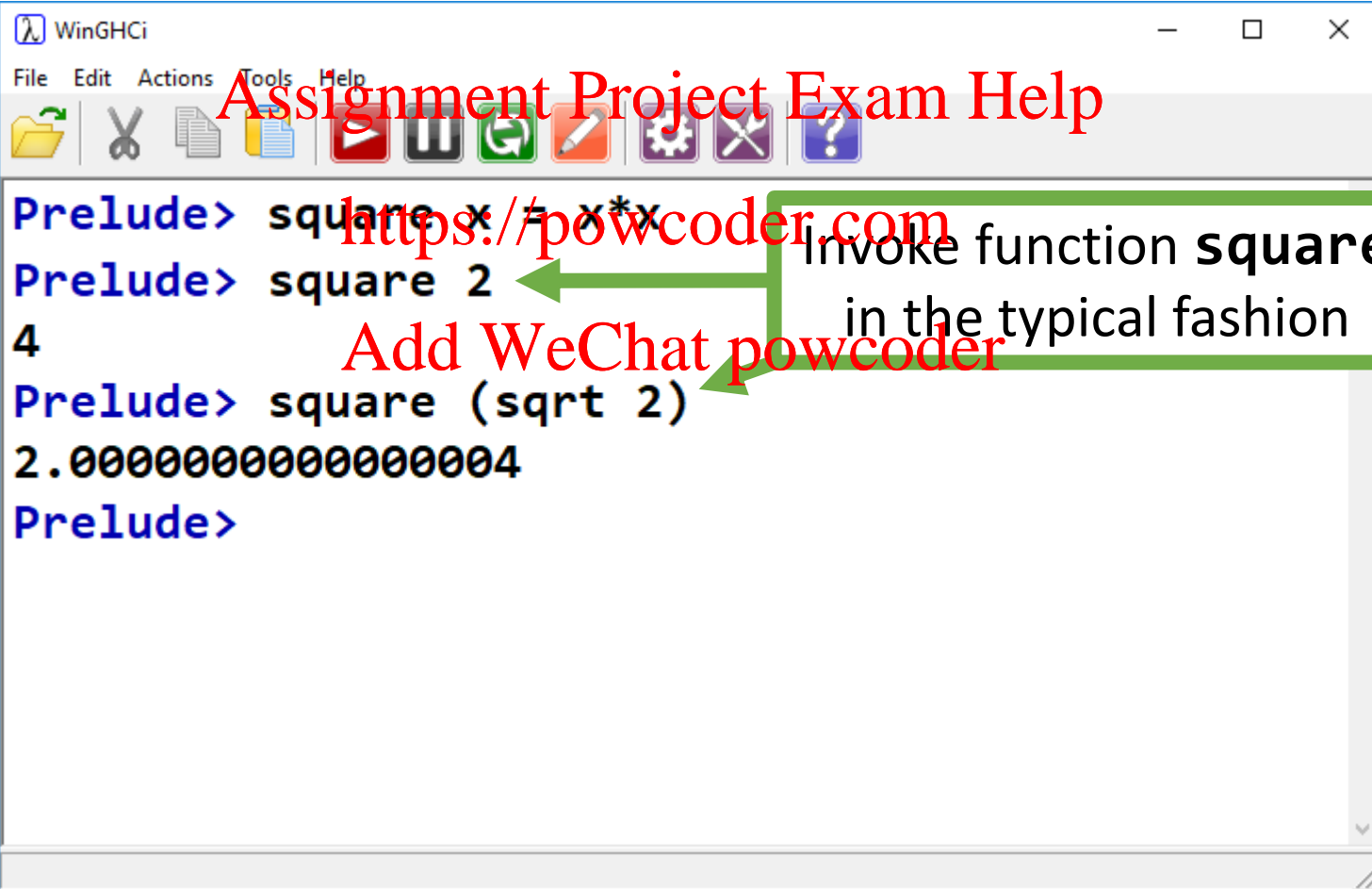
```
Prelude> square x = x*x
```

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

Square computes $x*x$, where
`x` is the input argument

Functions in Haskell

Let's start simple:



The screenshot shows the WinGHCi window with a menu bar (File, Edit, Actions, Tools, Help) and a toolbar. The command line contains the following text:

```
Prelude> square x = x*x
Prelude> square 2
4
Prelude> square (sqrt 2)
2.0000000000000004
Prelude>
```

Annotations on the image include:

- A red watermark "Assignment Project Exam Help" and the URL "https://powcoder.com" are overlaid on the top half of the window.
- A green box on the right contains the text "Invoke function **square** in the typical fashion".
- Two green arrows point from this box to the `square 2` and `square (sqrt 2)` lines in the code.
- Red text "Add WeChat powcoder" is located below the first arrow.

Functions in Haskell

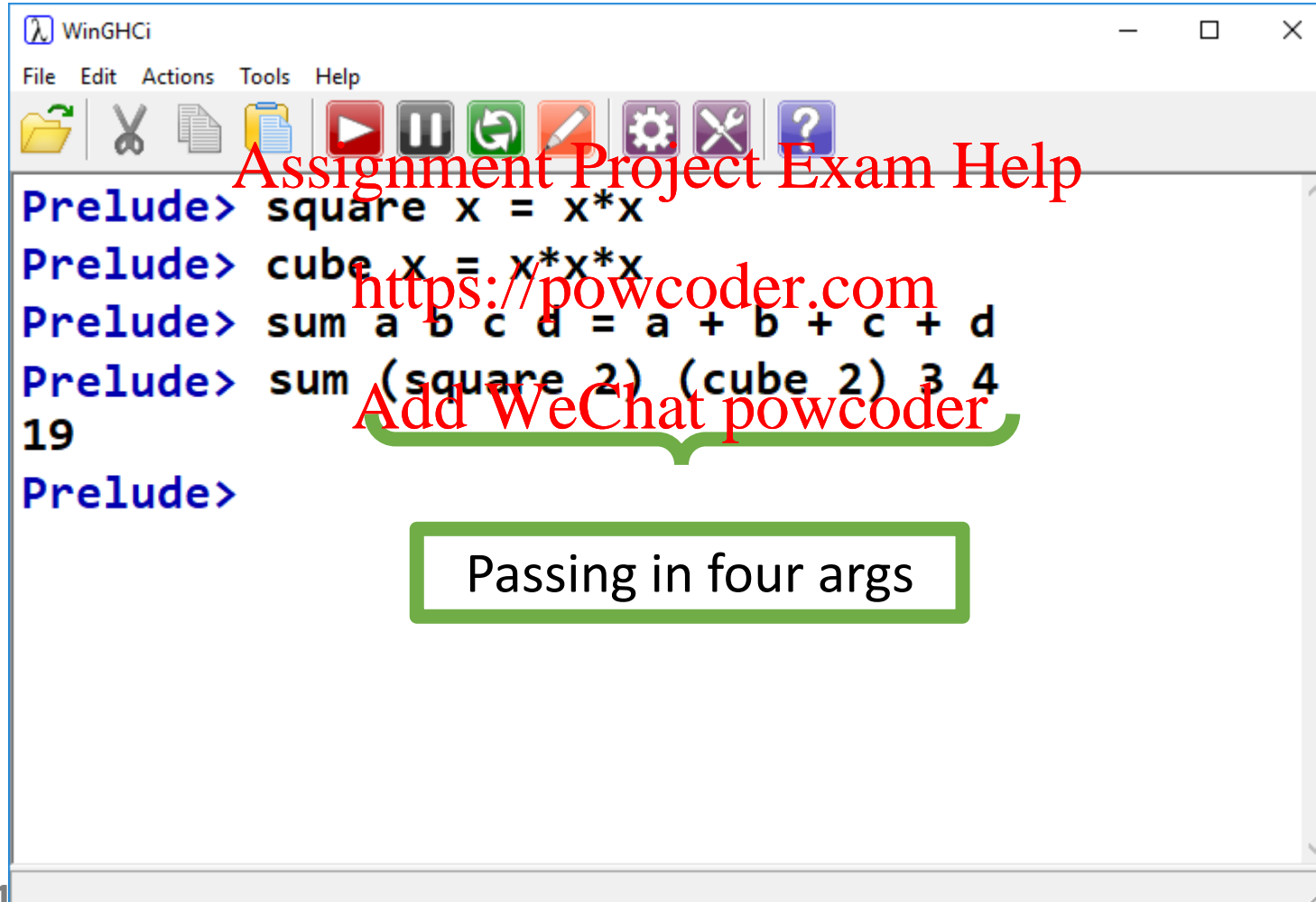
The screenshot shows the WinGHCi window with the following content:

```
WinGHCi
File Edit Actions Tools Help
[Icons]
Prelude> square x = x*x
Prelude> cube x = x*x*x
Prelude> sum a b c d = a + b + c + d
Prelude>
```

Annotations on the image:

- A red watermark "Assignment Project Exam Help" is at the top.
- A red watermark "https://powcoder.com" is in the center.
- A red watermark "Add WeChat powcoder" is at the bottom.
- A blue box labeled "Function named **sum**" has an arrow pointing to the `sum` keyword.
- A green box labeled "Parameter list" is underlined under `a b c d`.
- An orange box labeled "Expression to evaluate" is underlined under `= a + b + c + d`.

Functions in Haskell



The screenshot shows the WinGHCi window with the following content:

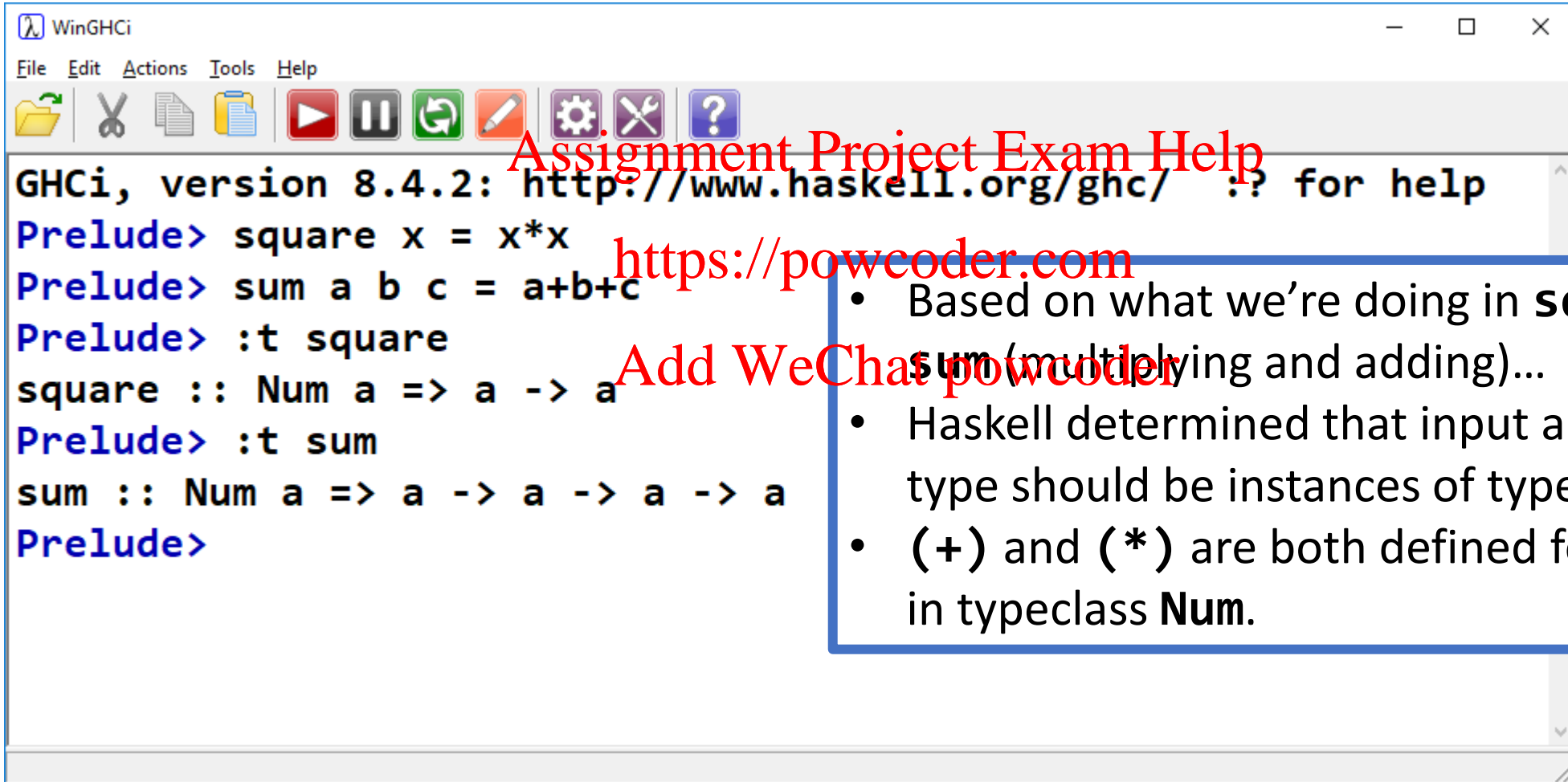
```
WinGHCi
File Edit Actions Tools Help
[Icons]
Prelude> square x = x*x
Prelude> cube x = x*x*x
Prelude> sum a b c d = a + b + c + d
Prelude> sum (square 2) (cube 2) 3 4
19
Prelude>
```

Overlaid on the screenshot are several red annotations:

- Assignment Project Exam Help** (top)
- <https://powcoder.com> (middle)
- Add WeChat powcoder** (bottom, underlined)

A green box highlights the text **Passing in four args**, which refers to the four arguments in the function call `sum (square 2) (cube 2) 3 4`.

Functions in Haskell



The screenshot shows the WinGHCi window with the following content:

```
WinGHCi
File Edit Actions Tools Help
[Icons]
GHCi, version 8.4.2: http://www.haskell.org/ghc/  :? for help
Prelude> square x = x*x
Prelude> sum a b c = a+b+c
Prelude> :t square
square :: Num a => a -> a
Prelude> :t sum
sum :: Num a => a -> a -> a -> a
Prelude>
```

Assignment Project Exam Help

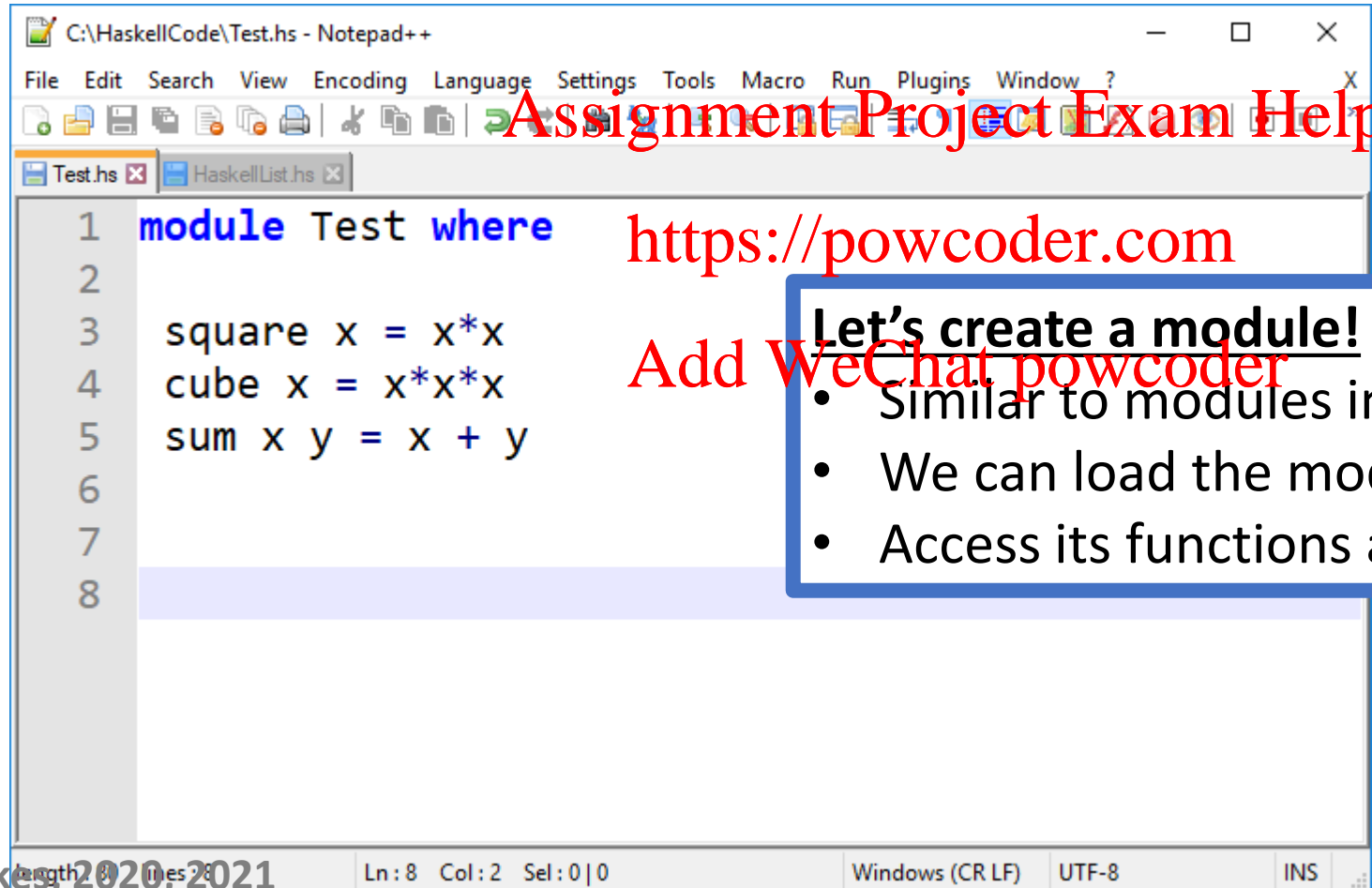
<https://powcoder.com>

Add WeChat powcoder

- Based on what we're doing in **square** and **sum** (multiplying 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.



The screenshot shows a Notepad++ window titled 'C:\HaskellCode\Test.hs - Notepad++'. The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Tools, Macro, Run, Plugins, and Window. The toolbar contains icons for file operations and editing. The code editor shows the following Haskell code:

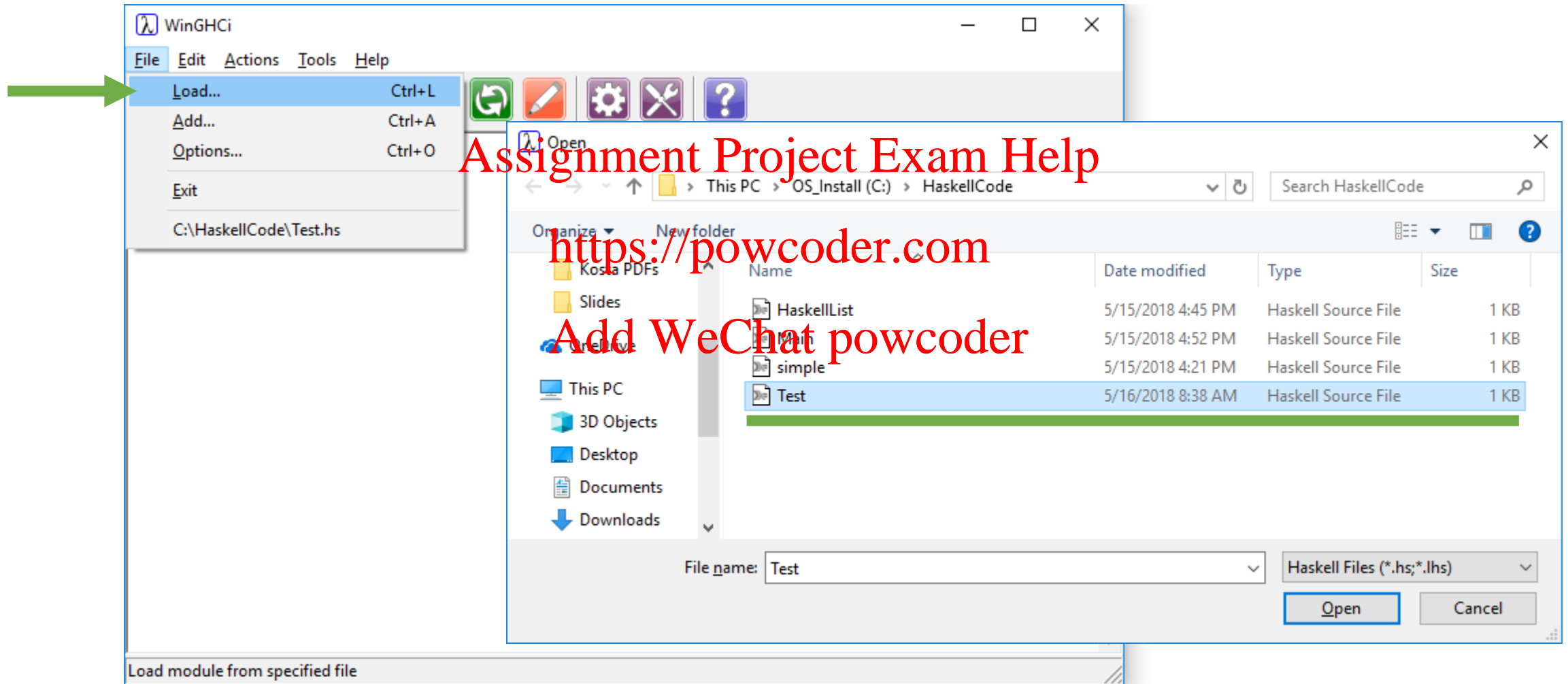
```
1 module Test where
2
3   square x = x*x
4   cube x = x*x*x
5   sum x y = x + y
6
7
8
```

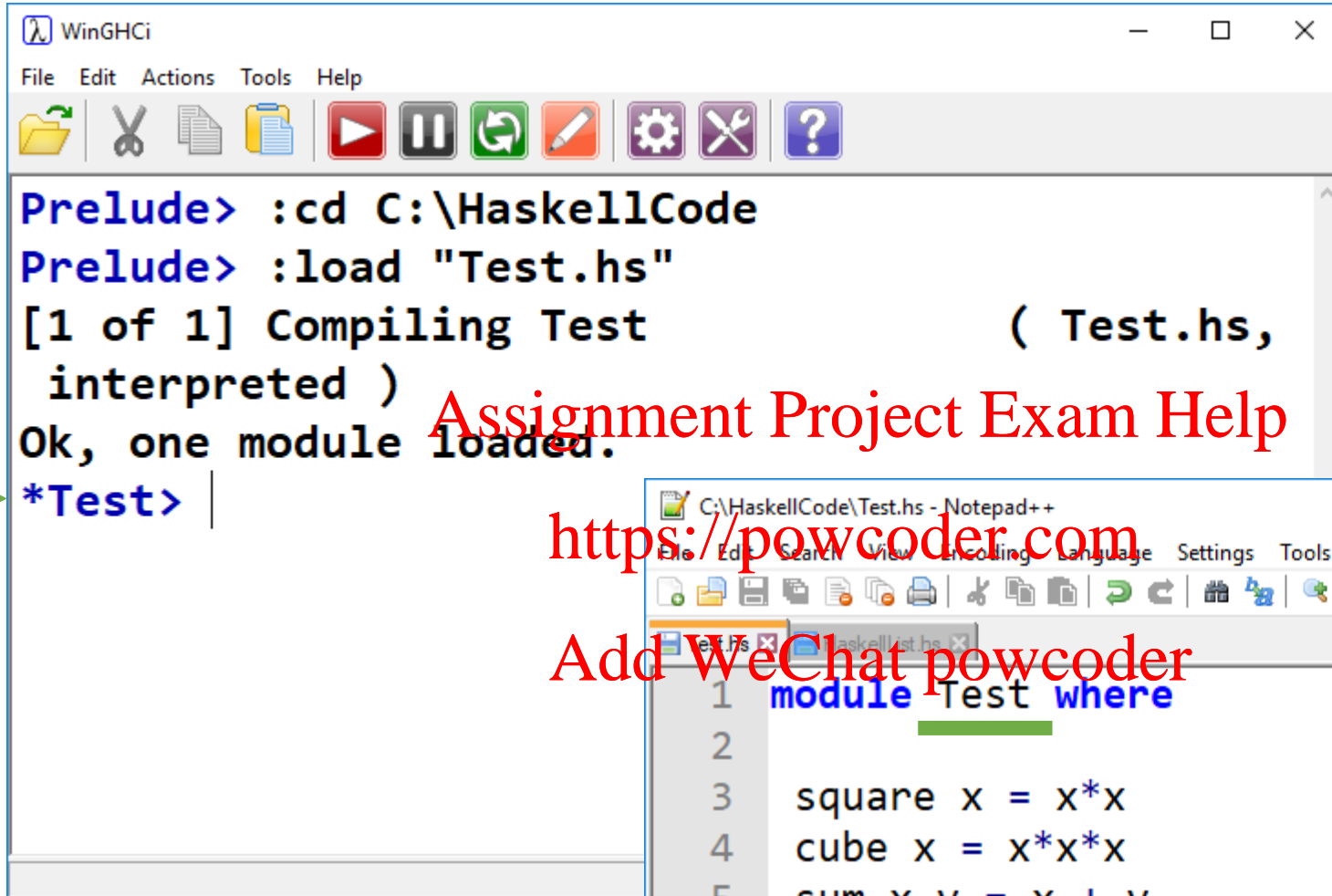
Overlaid on the screenshot is a red watermark that reads 'Assignment Project Exam Help' and a URL 'https://powcoder.com'. A blue-bordered box on the right contains the text 'Let's create a module!' followed by a bulleted list.

- Similar to modules in Elixir
- We can load the module in GHCi
- Access its functions and expressions

The status bar at the bottom shows 'Ln: 8 Col: 2 Sel: 0 | 0', 'Windows (CR LF)', 'UTF-8', and 'INS'.

Loading a Module





A screenshot of the WinGHCi terminal window. The window has a menu bar with 'File', 'Edit', 'Actions', 'Tools', and 'Help'. Below the menu is a toolbar with icons for file operations and execution. The terminal shows the following commands and output:

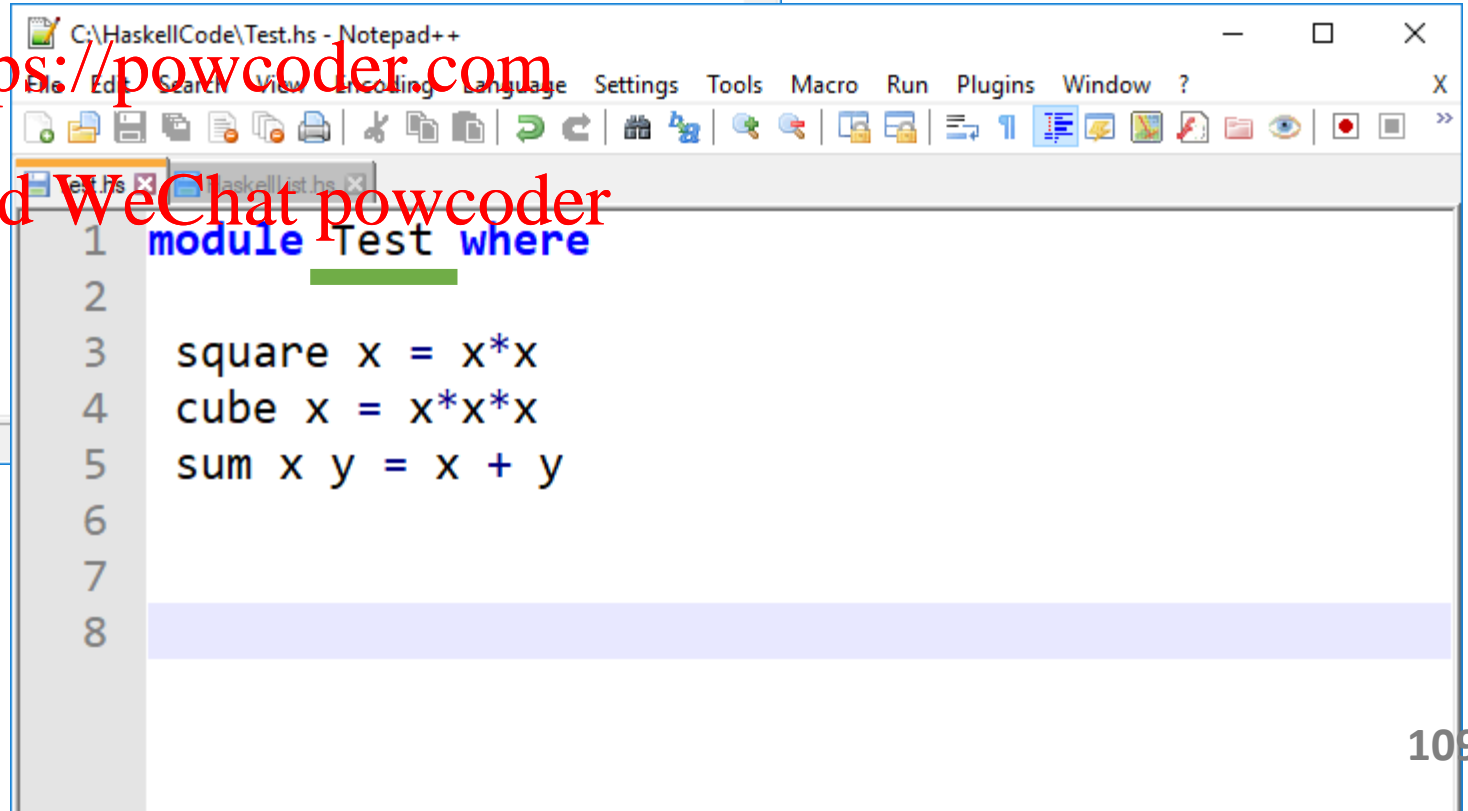
```
Prelude> :cd C:\HaskellCode
Prelude> :load "Test.hs"
[1 of 1] Compiling Test                ( Test.hs,
interpreted )
Ok, one module loaded.
*Test> |
```

A green arrow points to the prompt `*Test>`.

Assignment Project Exam Help

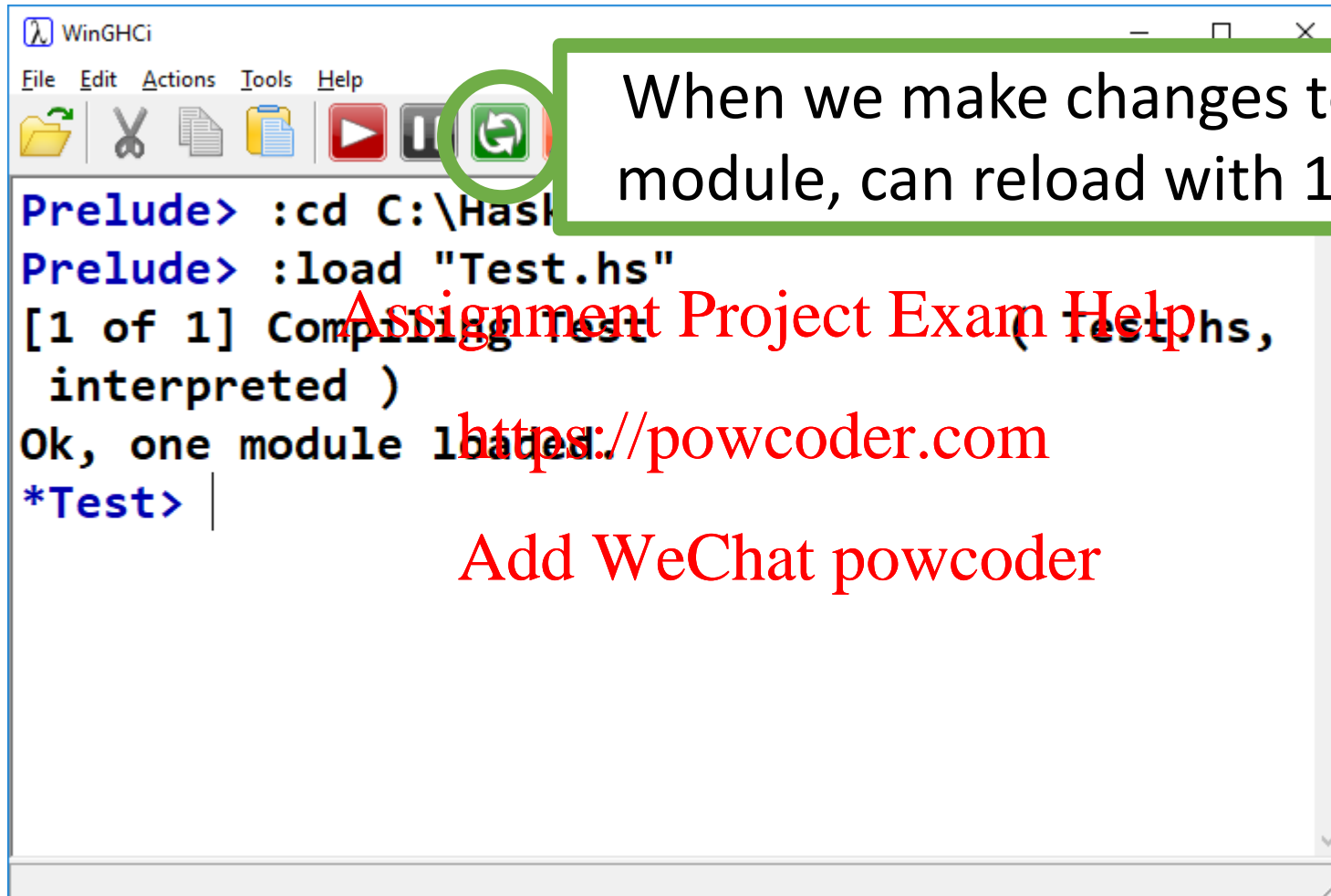
<https://powcoder.com>

Add WeChat powcoder



A screenshot of the Notepad++ editor window showing the contents of `C:\HaskellCode\Test.hs`. The window has a menu bar with 'File', 'Edit', 'Search', 'View', 'Encoding', 'Language', 'Settings', 'Tools', 'Macro', 'Run', 'Plugins', 'Window', and '?'. The toolbar includes icons for file operations, editing, and viewing. The code in the editor is as follows:

```
1 module Test where
2
3 square x = x*x
4 cube x = x*x*x
5 sum x y = x + y
6
7
8
```



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

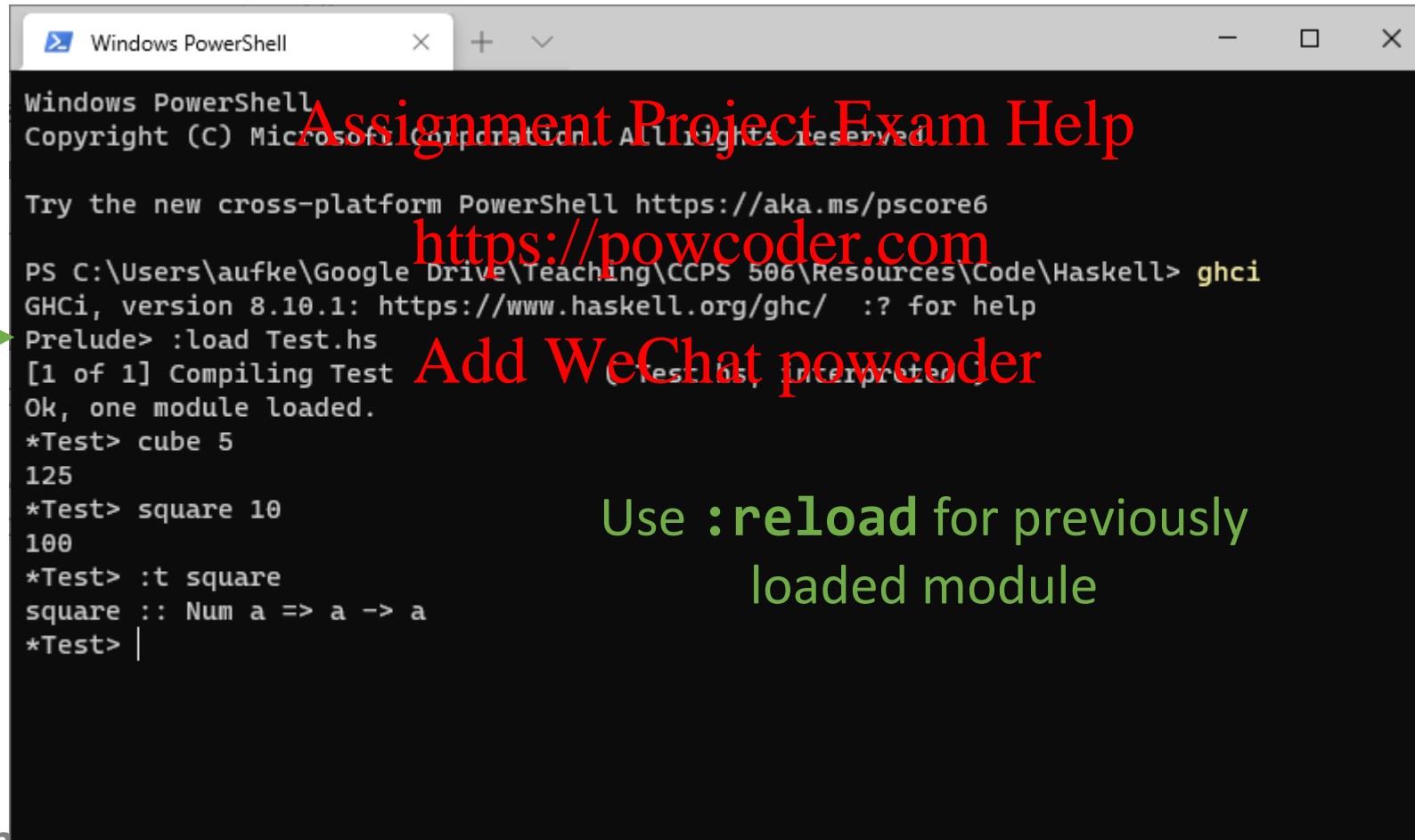
Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Loading a Module

Use `:load` in terminal GHCi:



The screenshot shows a Windows PowerShell terminal window with the following text:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

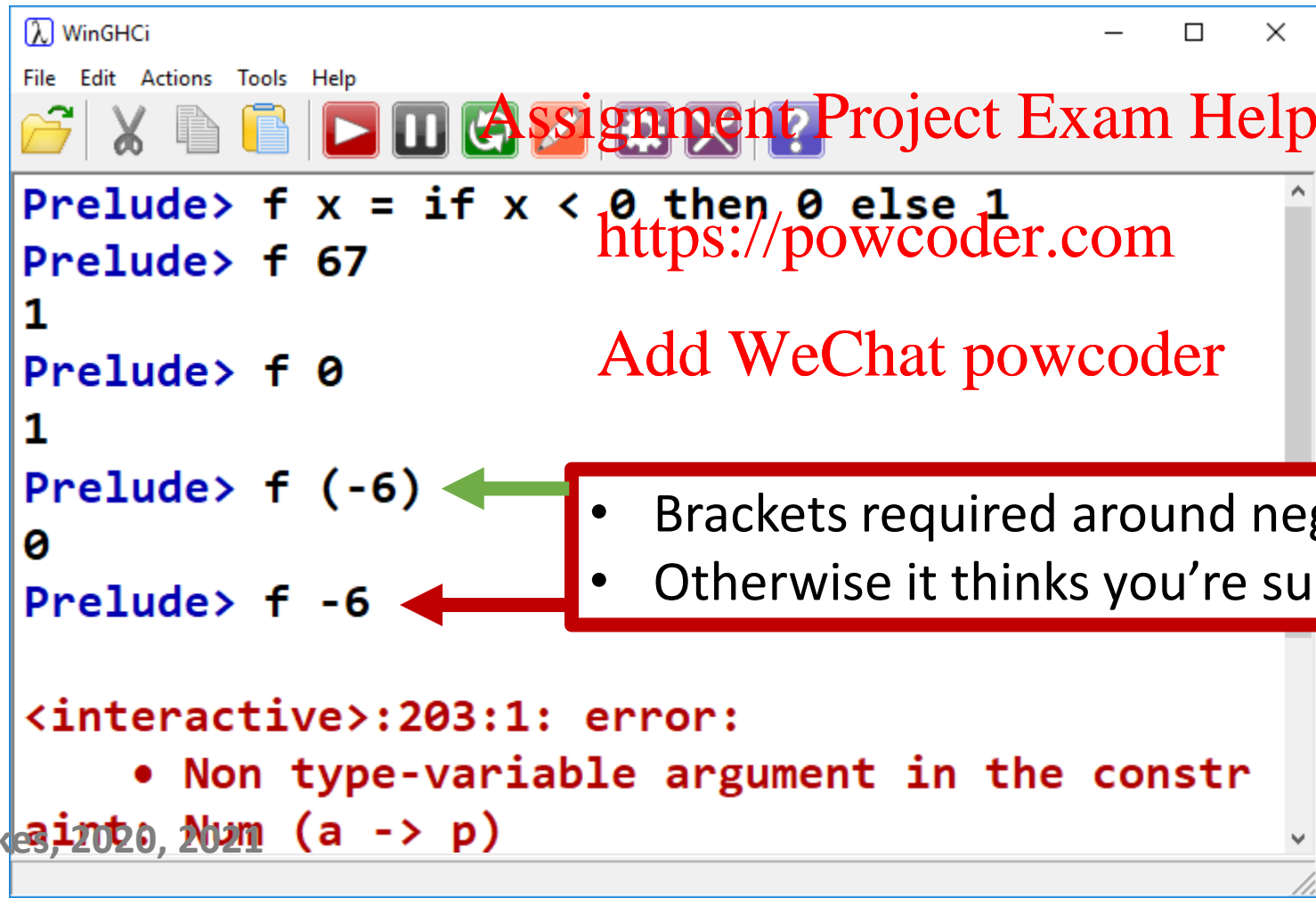
Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\aufke\Google Drive\Teaching\CCPS 506\Resources\Code\Haskell> ghci
GHCi, version 8.10.1: https://www.haskell.org/ghc/  :? for help
Prelude> :load Test.hs
[1 of 1] Compiling Test (Test.hs, interpreted)
Ok, one module loaded.
*Test> cube 5
125
*Test> square 10
100
*Test> :t square
square :: Num a => a -> a
*Test> |
```

A green arrow points to the `:load Test.hs` command. A green text box on the right says "Use `:reload` for previously loaded module".

Control Structures

if then else



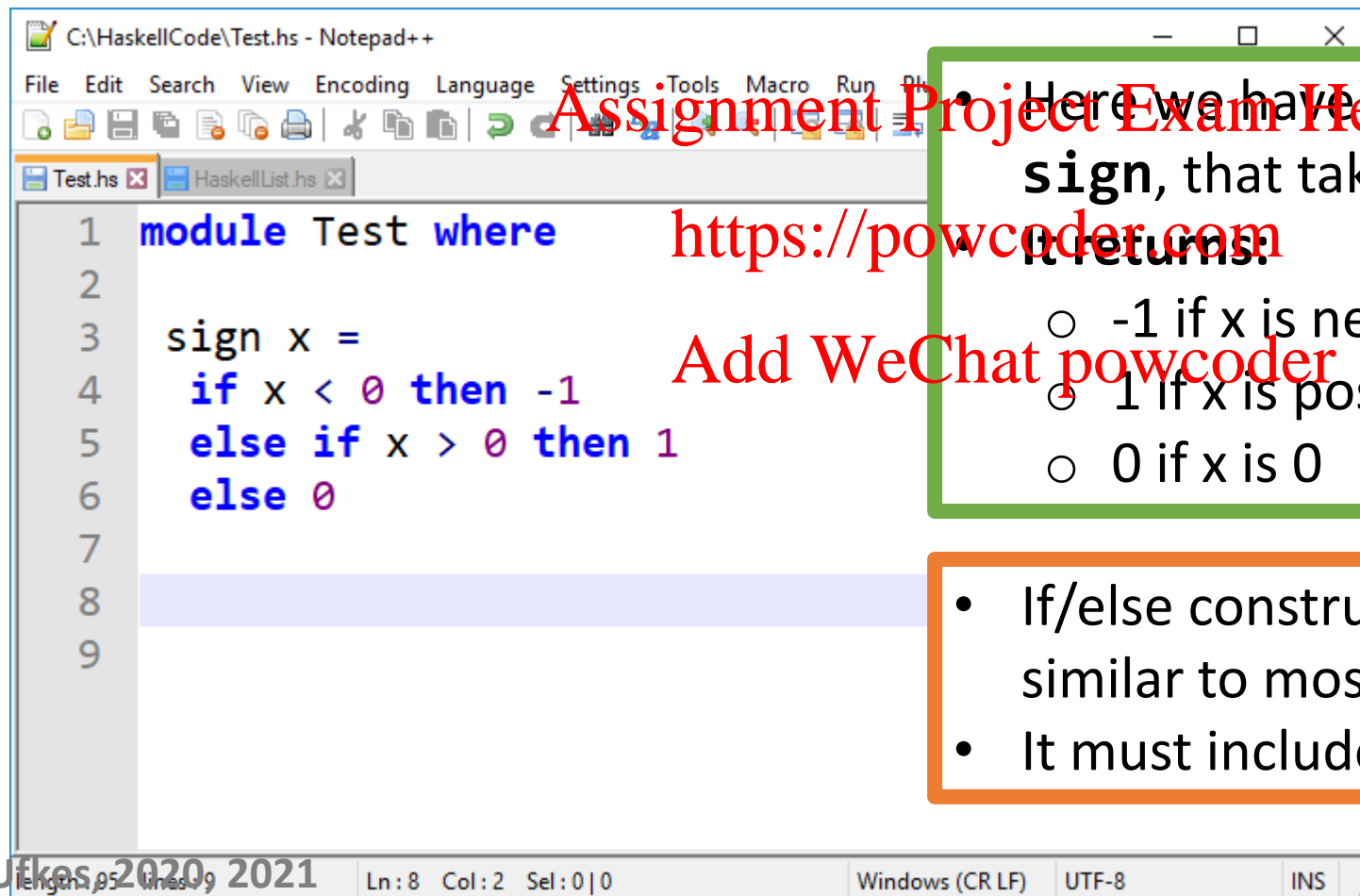
Assignment Project Exam Help
<https://powcoder.com>
Add WeChat powcoder

```
WinGHCi
File Edit Actions Tools Help
Prelude> f x = if x < 0 then 0 else 1
Prelude> f 67
1
Prelude> f 0
1
Prelude> f (-6)
0
Prelude> f -6
<interactive>:203:1: error:
  • Non type-variable argument in the constraint Num (a -> p)
    aint: Num (a -> p)
```

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

Control Structures

if then else if then else



The screenshot shows a Notepad++ window titled 'C:\HaskellCode\Test.hs - Notepad++'. The code is as follows:

```
1 module Test where
2
3 sign x =
4   if x < 0 then -1
5   else if x > 0 then 1
6   else 0
7
8
9
```

At the bottom of the window, the status bar shows 'Ln: 8 Col: 2 Sel: 0 | 0', 'Windows (CR LF)', 'UTF-8', and 'INS'.

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

- Here we have a function named **sign**, that takes one argument **x**

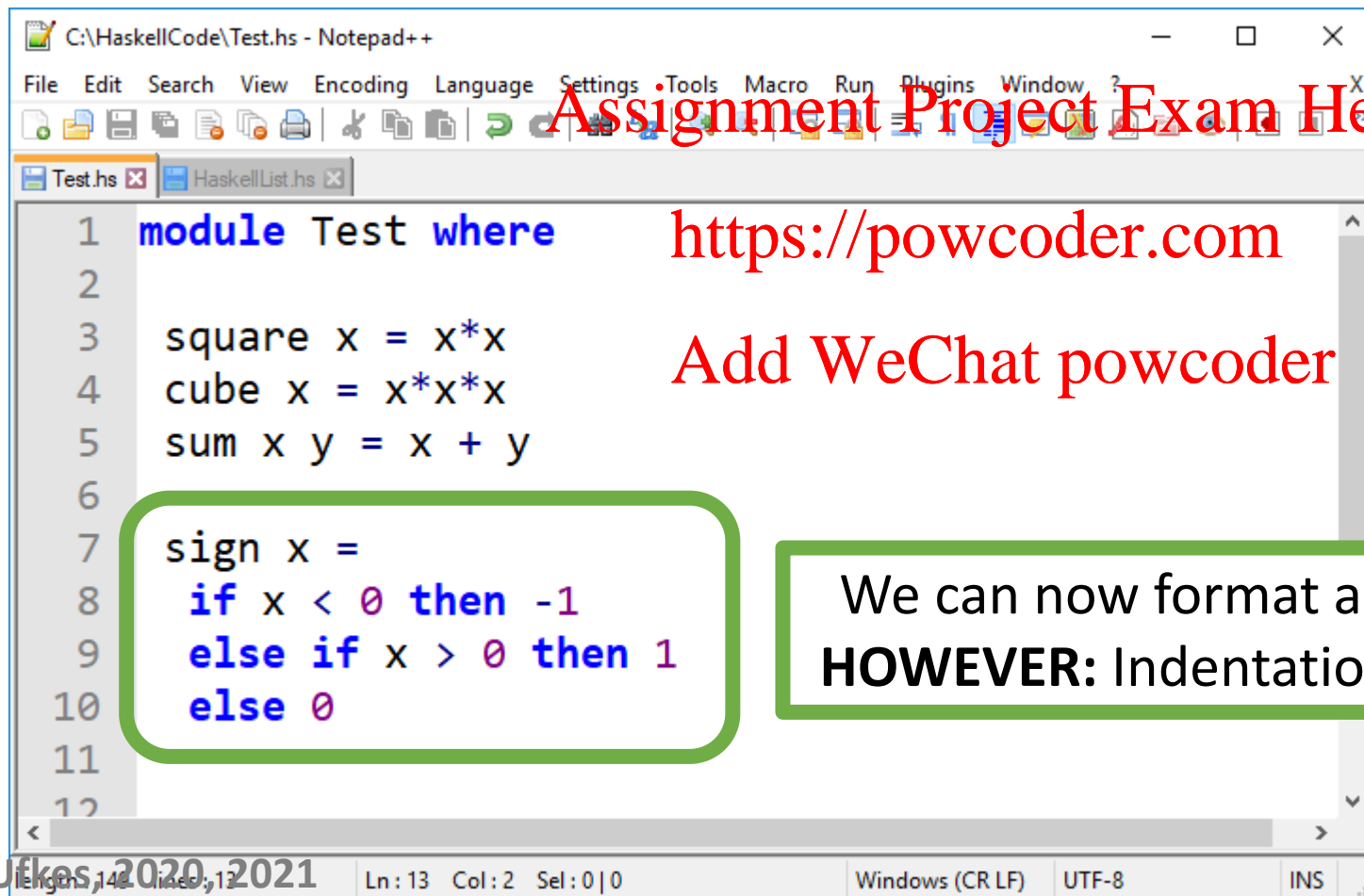
- It returns:

- -1 if x is negative
- 1 if x is positive
- 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



```
1 module Test where
2
3 square x = x*x
4 cube x = x*x*x
5 sum x y = x + y
6
7 sign x =
8   if x < 0 then -1
9   else if x > 0 then 1
10  else 0
11
12
```

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

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

```
C:\HaskellCode\Test.hs - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

Test.hs HaskellList.hs

1 module Test where
2
3 sign x =
4   if x < 0 then -1
5   else if x > 0 then 1
6   else 0
7
8
9

length: 92 lines: 9 Ln: 6 Col: 2 Sel: 0|0
```

```
WinGHCi
File Edit Actions Tools Help

Prelude> :load Test.hs
[1 of 1] Compiling Test
Failed, no modules loaded.
Prelude>
```

Assignment Project Exam Help

<https://powcoder.com>

Test.hs:4:2: error:
parse error (possibly incorrect indentation
or mismatched brackets)

```
4 | if x < 0 then -1 | ^
[1 of 1] Compiling Test (Test.hs,
interpreted )
Failed, no modules loaded.
Prelude>
```

```
C:\HaskellCode\Test.hs - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Test.hs HaskellList.hs

1 module Test where
2
3 sign x =
4   if x < 0 then -1
5   else if x > 0 then 1
6   else 0
7
8
9

length: 95 lines: 9 Ln: 4 Col: 3 Sel: 0|0
```

Assignment Project Exam Help

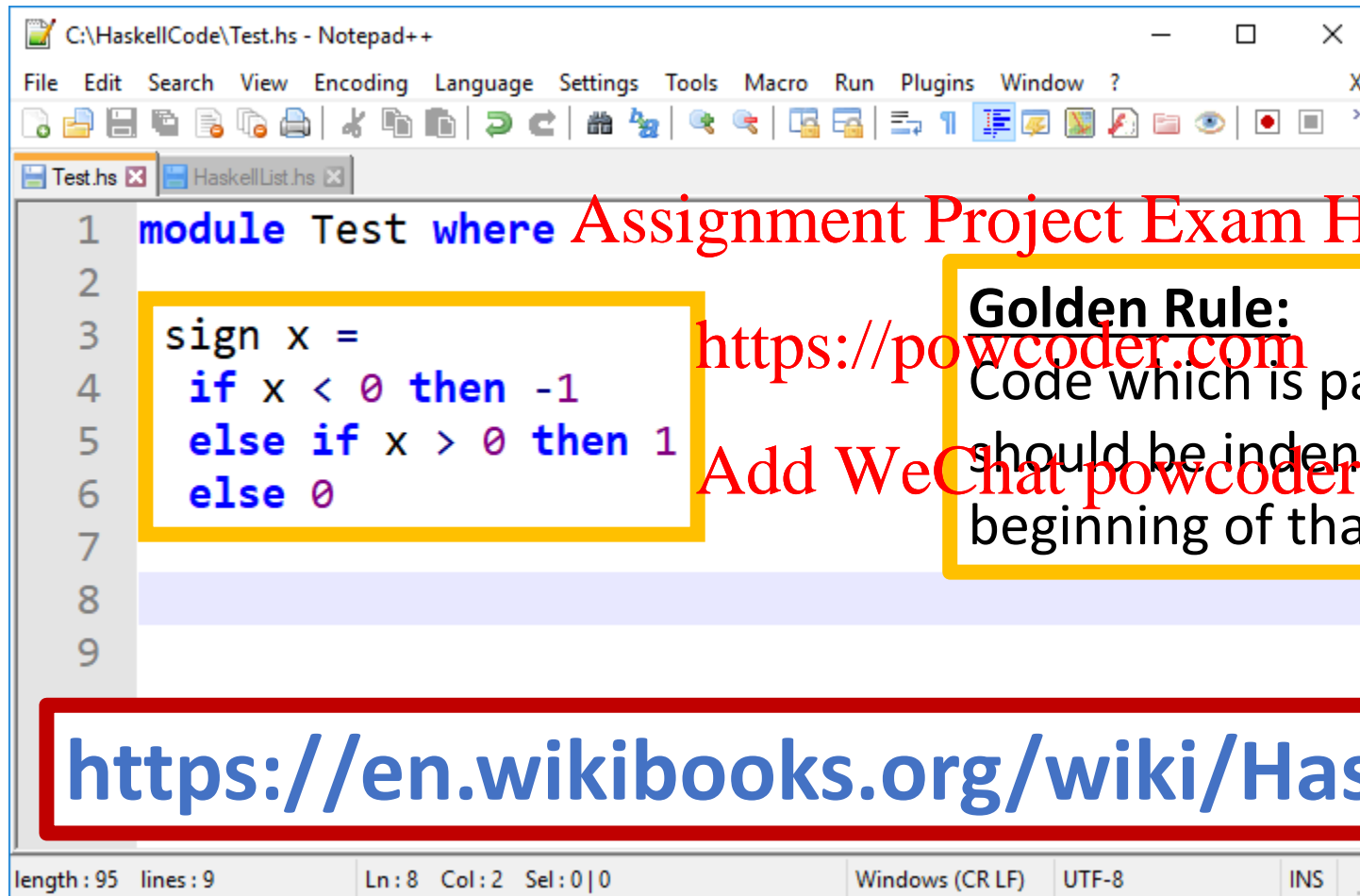
<https://powcoder.com>

Add WeChat powcoder

```
WinGHCi
File Edit Actions Tools Help
[1 of 1] Compiling Test (Test.hs,
Failed, no modules loaded.

Prelude> :reload
[1 of 1] Compiling Test (Test.hs,
interpreted )
Ok, one module loaded.
*Test>
```

Indenting in Haskell



The screenshot shows a Notepad++ window titled 'C:\HaskellCode\Test.hs - Notepad++'. The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Tools, Macro, Run, Plugins, Window, and ?. The toolbar contains various icons for file operations and editing. The code editor shows two tabs: 'Test.hs' and 'HaskellList.hs'. The code in 'Test.hs' is as follows:

```
1 module Test where
2
3 sign x =
4     if x < 0 then -1
5     else if x > 0 then 1
6     else 0
7
8
9
```

The status bar at the bottom indicates 'length : 95 lines : 9', 'Ln : 8 Col : 2 Sel : 0 | 0', 'Windows (CR LF)', 'UTF-8', and 'INS'.

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Golden Rule:

Code which is part of some expression should be indented further than the beginning of that expression

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

If all that weren't enough, Tabs don't work properly unless they're 8 spaces exactly.

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Haskell Tutorials/References:

https://en.wikibooks.org/wiki/Yet_Another_Haskell_Tutorial

Assignment Project Exam Help

<https://powcoder.com>

<http://cheatsheet.codeslower.com/CheatSheet.pdf>

Add WeChat powcoder

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

