COMP302: Programming Languages and Paradigms

Assignment Project Exam Help

Prof. Brigitte Pientka (Sec 01)
bpientka@cs.mcgill.ca

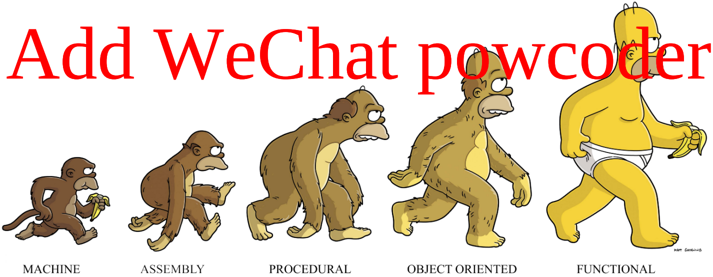
Francisco Ferreira (Sec 02)
fferre8@cs.mcgill.ca

https://powcoder.com

School of Computer Science
McGill University

Week 3-3, Fall 2017

Add WeChat powcoder



Functional Tidbit: Words of Wisdom

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

"Higher-order functions are super cool!"
- Eric Zhang (TA for COMP 302)

Why are higher-order functions cool?

Higher-order functions allow us to abstract over common functionality.

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Why are higher-order functions cool?

Higher-order functions allow us to abstract over common functionality.

Assignment Project Exam Help

- Programs can be very short and compact
- Programs are reusable, well-structured, modular!
- Each significant piece of functionality is implemented in one place.

https://powcoder.com

Add WeChat powcoder

Slogan

Functions are first-class values!

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Slogan

Functions are first-class values!

Assignment Project Exam Help

- Pass functions as arguments (Today)
- Return them as results (Next week)

https://powcoder.com

Add WeChat powcoder

Slogan

Functions are first-class values!

Assignment Project Exam Help

- Pass functions as arguments (Today)
- Return them as results (Next week)

https://powcoder.com

Add WeChat powcoder

Abstracting over common functionality

$$\sum_{k=a}^{k=b} k$$

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Abstracting over common functionality

$$\sum_{k=a}^{k=b} k$$

```
let rec sum (a,b) =
if a > b then 0 else a + sum(a+1,b)
```

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Abstracting over common functionality

$$\sum_{k=a}^{k=b} k$$

```
let rec sum (a,b) =
if a > b then 0 else a + sum(a+1,b)
```

Assignment Project Exam Help

$$\sum_{k=a}^{k=b} k^2$$

https://powcoder.com

Add WeChat powcoder

Abstracting over common functionality

$$\sum_{k=a}^{k=b} k$$

```
let rec sum (a,b) =
if a > b then 0 else a + sum(a+1,b)
```

Assignment Project Exam Help

$$\sum_{k=a}^{k=b} k^2$$

```
let rec sum (a,b) =
if a > b then 0 else square(a) + sum(a+1,b)
```

https://powcoder.com

$$\sum_{k=a}^{k=b} 2^k$$

Add WeChat powcoder

Abstracting over common functionality

$$\sum_{k=a}^{k=b} k$$

```
let rec sum (a,b) =
if a > b then 0 else a + sum(a+1,b)
```

Assignment Project Exam Help

$$\sum_{k=a}^{k=b} k^2$$

```
let rec sum (a,b) =
if a > b then 0 else square(a) + sum(a+1,b)
```

https://powcoder.com

$$\sum_{k=a}^{k=b} 2^k$$

```
let rec sum (a,b) =
if a > b then 0 else exp(2,a) + sum(a+1,b)
```

Add WeChat powcoder

Abstracting over common functionality

$$\sum_{k=a}^{k=b} k$$

```
let rec sum (a,b) =
if a > b then 0 else a + sum(a+1,b)
```

Assignment Project Exam Help

$$\sum_{k=a}^{k=b} k^2$$

```
let rec sum (a,b) =
if a > b then 0 else square(a) + sum(a+1,b)
```

https://powcoder.com

$$\sum_{k=a}^{k=b} 2^k$$

```
let rec sum (a,b) =
if a > b then 0 else exp(2,a) + sum(a+1,b)
```

Add WeChat powcoder

Can we write a generic sum function?

Non-Generic Sum (old)	Generic Sum using a function as an argument
`sum: int * int -> int`	`sum: (int -> int) -> int * int ->  int`

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Demo

Abstracting over common functionality

```
let rec sum f (a, b) =
if (a > b) then 0 else (f a) + sum f (a+1, b)
```

Assignment Project Exam Help

How about only summing up odd numbers between a and b?

https://powcoder.com

Add WeChat powcoder

Abstracting over common functionality

```
let rec sum f (a, b) =
if (a > b) then 0 else (f a) + sum f (a+2, b)
```

How about only summing up even numbers between `a` and `b`?

```
let rec sumOdd (a, b) =
 if (a mod 2) = 1 then
  sum (fun x -> x) (a, b)        (* a was odd *)
 else
  sum (fun x -> x) (a+1, b)      (* a was even *)
```

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

Abstracting over common functionality (increment)

```
let rec sum f (a, b) inc =
if (a > b) then 0 else (f a) + sum f (inc(a), b) inc
```

Assignment Project Exam Help

How about only summing up even numbers between a and b?

https://powcoder.com

Add WeChat powcoder

```
let rec sumOdd (a, b) =
 if (a mod 2) = 1 then
  sum (fun x -> x) (a, b) (fun x -> x + 2)        (* a was odd *)
 else
  sum (fun x -> x) (a+1, b) (fun x -> x + 2)      (* a was even *)
```

Abstracting over common functionality how we combine numbers in each step

Assignment Project Exam Help

```
let rec sum f (a, b) inc =
if (a > b) then 0 else (f a) + sum f (inc(a), b) inc
```

https://powcoder.com

How about only multiplying numbers between `a` and `b`?

Add WeChat powcoder

Abstracting over common functionality
how we combine numbers in each step

Assignment Project Exam Help

```
let rec sum f (a, b) inc =
if (a > b) then 0 else (f a) + sum f (inc(a), b) inc
```

https://powcoder.com

How about only multiplying numbers between `a` and `b`?

Add WeChat powcoder

```
let rec product f (a, b) inc =
if (a > b) then 1 else (f a) * product f (inc(a), b) inc
```

Abstracting over common functionality (tail-recursively) how we combine numbers in each step

Assignment Project Exam Help

```
let rec sum f (a, b) inc acc =
if (a > b) then 0 else sum f (inc(a), b) inc (f a + acc)
```

https://powcoder.com

How about only multiplying numbers between a and b?

Add WeChat powcoder

Abstracting over common functionality (tail-recursively) how we combine numbers in each step

Assignment Project Exam Help

```
let rec sum f (a, b) inc acc =
if (a > b) then 0 else sum f (inc(a), b) inc (f a + acc)
```

https://powcoder.com

How about only multiplying numbers between `a` and `b`?

Add WeChat powcoder

```
let rec product f (a, b) inc acc =
if (a > b) then 1 else product f (inc(a), b) inc (f a * acc)
```

Assignment Project Exam Help

Demo

https://powcoder.com

Add WeChat powcoder

Take away

Abstraction and higher-order functions are very powerful mechanisms for writing reusable programs.

Assignment Project Exam Help

Computing a series

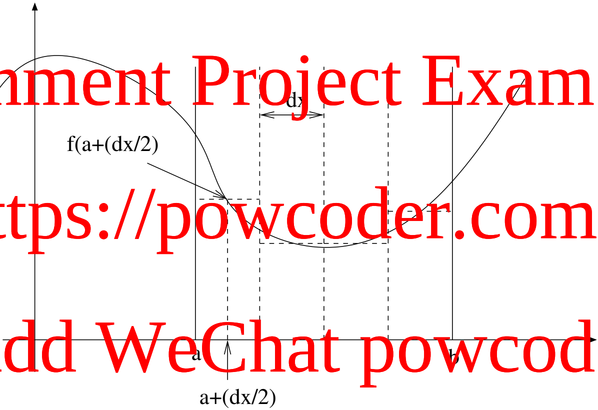
https://powcoder.com

```
series:  (int -> int -> int)    (*  comb    *)
      -> (int -> int)           (*  f       *)
      -> (int * int)            (*  (a,b)   *)
      -> (int -> int)           (*  inc     *)
      -> int                    (*  acc     *)
      -> int                    (*  result  *)
```

Add WeChat powcoder

```
let sum  f (a,b) inc = series (fun x y -> x + y) f (a,b) inc 0
let prod f (a,b) inc = series (fun x y -> x * y) f (a,b) inc 1
```

Bonus: Approximating the integral!



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

Let $I = a + dx/2$.

$$\begin{aligned}\int_a^b f(x)\,dx &\approx f(I) * dx + f(I + dx) * dx + f(I + dx + dx) * dx + \ldots \\ &= dx * (f(I) + f(I + dx) + f(I + 2 * dx) + f(I + 3 * dx) \ldots)\end{aligned}$$

Higher-order functions on data types

More higher-order functions next week!

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder