

CS 160 Compilers

程序代写代做 CS编程辅导



Lecture 2: OCaml Crash Course I

Assignment Project Exam Help

Email: tutorcs@163.com

QQ: 749389476

<https://tutorcs.com>

Yu Feng
Spring 2023

Introducing the east

程序代写代做CS编程辅导

Sections:



- Thur 3 (ILP 209)
- 4-5pm (ILP 4209)
- 5-6pm (ILP 2207)

WeChat: cstutorcs

Assignment Project Exam Help

Instructor's office hour: Wed, 3-4pm, HHH 2157

Email: tutorcs@163.com

TA's office hour:

QQ: 749389476

- Jingtao Xia: Tue 1:30-2:30pm (Phelps 3523)
- Junrui Liu: Thur 11am-12pm (Phelps 3523)
- Thanawat Techaumnuauiwit: Fri 1-2pm (CSIL)

<https://tutorcs.com>

History of ML



- ML = “Meta Language”
- Designed by Robin Milner @ Edinburgh
- Language to manipulate Theorems/Proofs
- Several dialects:
 - “Standard” ML (of New Jersey)
 - French dialect with support for objects
 - State-of-the-art
 - Extensive library, tool, user support

WeChat: cstutorcs

Assignment Project Exam Help

Email: tutorcs@163.com

QQ: 749389476

<https://tutorcs.com>



Who are using OCaml

程序代写代做CS编程辅导



WeChat: cstutorcs



docker

Assignment Project Exam Help

Email: tutorcs@163.com

Bloomberg

QQ: 749389476



Jane Street

<https://tutorcs.com>

Ocaml vs. C

程序代写代做 CS编程辅导



```
void sort(int arr[], int beg, int end) {
    if (end > beg + 1) {
        int piv = arr[beg];
        int l = beg + 1;
        int r = end;
        while (l != r-1) {
            if(arr[l] <= piv)
                l++;
            else
                swap(&arr[l], &arr[r--]);
        }
        if(arr[l]<=piv && arr[r]<=piv)
            l=r+1;
        else if(arr[l]<=piv && arr[r]>piv)
            {l++; r--;}
        else if (arr[l]>piv && arr[r]<=piv)
            swap(&arr[l++], &arr[r--]);
        else
            r=l-1;
        swap(&arr[r--], &arr[beg]);
        sort(arr, beg, r);
        sort(arr, l, end);
    }
}
```

Quicksort in C

```
let rec sort l =
  match l with [] -> []
  | h::t ->
    let (l,r)= List.partition ((<=) h) t in
    (sort l)@h::(sort r)
```

Quicksort in Ocaml

WeChat: cstutorcs

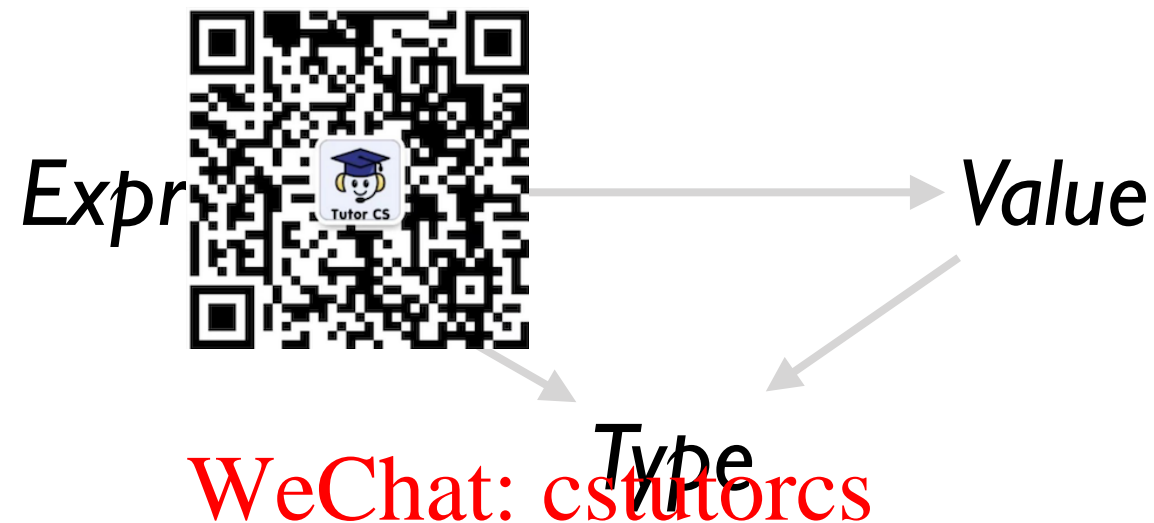
Assignment Project Exam Help

Email: tutorcs@163.com

QQ: 749389476

<https://tutorcs.com>

ML's holy grail



Assignment Project Exam Help

- Email: tutorcs@163.com
 - Everything has a value
 - Everything has a type
- <https://tutorcs.com>

Interacting with ML



val-Print” Loop

Repeat:


1. System reads expression e
2. System evaluates e to get value v
3. System prints value v and type t

What are these expressions, values and types ?

<https://tutorcs.com>

Basic types

程序代写代做CS编程辅导

# 2;			2
# 2+3;;		Int	5

WeChat: cstutorcs

Assignment Project Exam Help

Email: tutorcs@163.com

# "hi,"^"0Caml";;	String	"hi, 0Caml"
-------------------	--------	-------------

QQ: 749389476

# true;;		true
----------	--	------

<https://tutorcs.com>

# 2>3;;	Bool	false
---------	------	-------

Type errors

程序代写代做CS编程辅导



```
# "Hi," ^ 2;;  
# (2+3) || 9;;
```

Assignment Project Exam Help

Untypable expression is rejected

- No casting or coercing
- Fancy algorithm to catch errors
- ML's single most powerful feature

Email: tutorcs@163.com

QQ: 749389476

<https://tutorcs.com>

Complex types: Lists

程序代写代做CS编程辅导



List operators:

- Cons (::): “cons” an element to a list of same type
- append (@): only append two list of the same type
- Head (List.hd): return the head element of a nonempty list
- Tail (List.tl): return the tail of nonempty list

WeChat: cstutors

Assignment Project Exam Help

Syntax:

Email: tutorcs@163.com

- Lists = semicolon

QQ: 749389476

Semantics:

<https://tutorcs.com>

- Same type, unbounded number

Complex types: Tuples



```
# (9-3, "ab" ^ "cd" ^ ">8") ; ;
```

```
(6, "abcd", (4, false))
```

WeChat: cstutorcs

int * string * (int * bool)

Assignment Project Exam Help

Email: tutorcs@163.com

QQ: 749389476

<https://tutorcs.com>

Syntax:

- Lists = comma

Semantics:

- Different type, fixed number

Variables and bindings



$x = e$

“Bind the value of expression e to the variable x ”

WeChat: cstutorcs

Assignment Project Exam Help

Email: tutorcs@163.com

val x : int = 4

QQ: 749389476

<https://tutorcs.com>

Variables and bindings



Later declare expressions can use x

- Most recent value used for evaluation

```
# let x = 2+2;;  
val x : int = 4  
# let y = x * x * x;;  
val y : int = 64  
# let z = [x;y;x+y];;  
val z : int list = [4;64;68]
```

<https://tutorcs.com>

Variables and bindings



Undeclared variable (i.e. without a value binding)
it accepted !

```
# let p = a + 1;  
Characters 8-9:  
    let p = a + 1 ;;  
^ Unbound value a
```

QQ: 749389476

<https://tutorcs.com>

Local bindings



for expressions using “temporary” variables

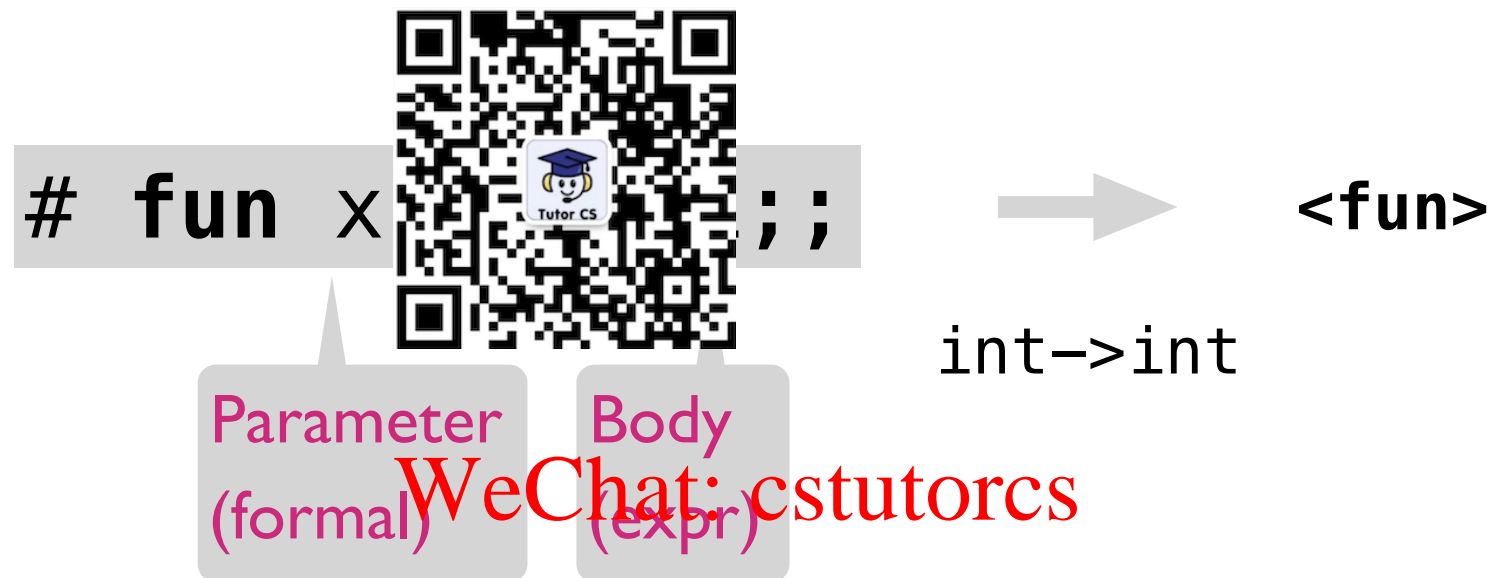
```
# let  
  tempVar = x + 2 * y  
in  
  tempVar * tempVar ;;
```

WeChat: cstutorcs
Assignment Project Exam Help
Email: tutorcs@163.com

QQ: 749389476

- tempVar is bound only inside expr body from in ... ;;
- Not visible (in scope) outside

Complex types: functions



WeChat: cstutorcs

Assignment Project Exam Help

```
# let inc = fun x -> x+1;
val inc : int -> int = fn
# inc 0;
val it : int = 1
# inc 10;
val it : int = 11
```

Email: tutorcs@163.com

QQ: 749389476

<https://tutorcs.com>

How to evaluate a function app:

- Evaluate the argument
- Bind formal to arg value
- Evaluate the "body expr"

Complex types: functions



fun x -> fun <y;;



<fun>

'a->('a->bool)

WeChat: cstutorcs

Wow! A function can return a function
Assignment Project Exam Help

```
# let lt = fun x -> fun y -> x < y;;  
val lt : 'a -> 'a -> bool = fn  
# let is5lt = lt 5;  
val is5lt : int -> bool = fn;;  
# is5lt 10;;  
val it : bool = true;  
# is5lt 2;;  
val it : bool = false;
```

Email: tutorcs@163.com

QQ: 749389476

<https://tutorcs.com>

Complex types: functions



fun f -> fun x (f x);; → <fun>

('a->bool)->('a->bool)

WeChat: cstutorcs

A function can also take a function argument

Assignment Project Exam Help

```
# let neg = fun f -> fun x -> not (f x);
val lt : (a -> bool) -> a -> bool = fn
# let is5gte = neg is5lt;
val is5gte : int -> bool = fn
# is5gte 10;
val it : bool = false;
# is5gte 2;
val it : bool = true;
```

Email: tutorcs@163.com

QQ: 749389476

<https://tutorcs.com>

Pattern matching



A pattern matching is what similar to switch statement but offers a lot more expressive power. It really boils down to matching an argument against an exact value, a predicate, or a type constructor.

WeChat: cstutorcs

```
type animal = Dog of string | Cat of string ;;
```

```
let say x =  
  match x with  
  | Dog x -> x ^ " says woof"  
  | Cat x -> x ^ " says meow"  
;;  
  
say (Cat "Tom") ;; (* "Tom says meow". *)
```

Email: tutorcs@163.com

QQ: 749389476

<https://tutorcs.com>

Put it together: a “filter” function

If arg matches
this pattern



Use this
expr

```
# let rec filter f l =  
  match l with  
  [] -> []  
  | (h::t) -> if f h then h::(filter f t)  
                else (filter f t);;
```

```
val filter : ('a -> bool) -> 'a list -> 'a list = <fun>
```

```
# let list1 = [1;31;12;4;7;2;10];;  
# filter is5lt list1 ;;
```

```
val it : int list = [31;12;7;10]
```

WeChat: cstutorcs

Assignment Project Exam Help

Email: tutorcs@163.com

QQ: 749389476

<https://tutorcs.com>

Put it together: a “quicksort” function



```
# let partition f l = filter f l, filter (neg f) l;;  
val partition : ('a -> bool) -> 'a list -> 'a list * 'a list = fn  
# let list1 = [1,31,12,4,7,2,10];  
# partition is5lt WeChat: cstutorcs  
val it : (int list * int list) = ([31,12,7,10], [1,2,10])
```

Assignment Project Exam Help

```
# let rec sort l =  
  match l with  
  [] -> []  
  | (h::t) -> QQ: 749389476  
    let (l,r) = partition ((<) h) t in  
    (sort l)@h::(sort r);;  
https://tutorcs.com
```

TODOs by next lecture



- Get familiar with (
- Come to the discussion session if you are new to OCaml

WeChat: cstutorcs

Assignment Project Exam Help

Email: tutorcs@163.com

QQ: 749389476

<https://tutorcs.com>