Welcome

CS1101S Studio Session Week 9: *Environment Model, Array & Loop*

Niu Yunpeng

niuyunpeng@u.nus.edu

October 16, 2018

Before We Start

Sumobot Contest

- Time: 6pm, 17 October 2018
- Venue: SR1, Level 2, COM1
- Free XP!
- Free food!

Overview

From Last Week

Immutable

- A constant holds a value inside it.
- `const x = 1;`
- Cannot hold another value.

Mutable

- A new value can be assigned to the same variable.
- `<variable_name> = <new_value>`
- `let y = 2;`
- To change the value inside `y = 3;`

From Last Week

Before Week 8

- Pure functional programming.
- Substitution model.
- Return value do not change if values of arguments are the same.

After Week 8

- Stateful programming.
- Environment model.
- Return value may vary even if values of arguments are the same.

Environment Model

Environment model

- Even though we supply the same values for all arguments, the return value of a function may still vary.
- Due to this, the substitution model breaks down.
- We have to introduce a new one and a better one:

  environment model
- It is an *upgrade* of substitution model + variable scoping.

Environment Model

Frame

- Each function call creates a new frame (similar to scope for variables).
- The initial frame is called global frame (global scope).
- Each frame contains a series of bindings of names and values.

Environment

- In order to find the variable, it is possible to search starting from the current local scope up to the global scope.
- Thus, all these corresponding frames are deterministic to the value of the variable. They are called the environment, a sequence of frames.

Environment Model

Frame & environment

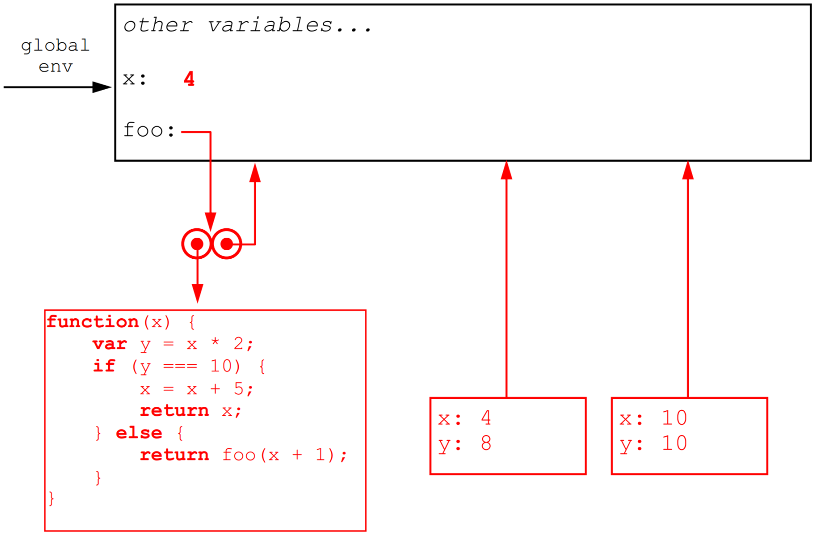
- Looks like a list.
- The head is the current frame, while the tail is pointing to the parent frames, called its **enclosing environment**.

Environment Model

What happens when we call a function?

- Create a new frame to extend the current environment.
- Evaluate actual arguments and bind their values to formal parameters.
- Local variables are bound to `undefined`.
- Evaluate the function body and send the return value to the enclosing environment.

Environment Model



```
function(x) {
    var y = x * 2;
    if (y === 10) {
        x = x + 5;
        return x;
    } else {
        return foo(x + 1);
    }
}
```

Environment Model

Exercise 1

- Assume the program stops at the comment.
- Draw the environment model diagram gradually.
- Also, identify the value of x at the point of that comment.

Environment Model

Exercise 1.1

```
let x = 0;

function environmentalist() {
    x = x + 1;

    function model(x) {
        x = x + 2;
        return x;
    }

    return model(x);
}
// Here
environmentalist();
x = environmentalist();
```

Environment Model

Exercise 1.2

```
let x = 0;

function environmentalist() {
    x = x + 1;

    function model(x) {
        x = x + 2;
        // Here
        return x;
    }

    return model(x);
}

environmentalist();
x = environmentalist();
```

Environment Model

Exercise 1.3

```
let x = 0;

function environmentalist() {
    x = x + 1;

    function model(x) {
        x = x + 2;
        return x;
    }

    return model(x);
}

environmentalist();
// Here
x = environmentalist();
```

Environment Model

Exercise 1.4

```
let x = 0;

function environmentalist() {
    x = x + 1;

    function model(x) {
        x = x + 2;
        return x;
    }

    return model(x);
}

environmentalist();
x = environmentalist();
// Here
```

Environment Model

Exercise 2

- The whole program has been evaluated.
- Draw the environment model diagram.

Environment Model

Exercise 2.1

```
let x = 4;

function foo(x) {
    let y = x * 2;

    if (y === 10) {
        x = x + 5;
        return x;
    } else {
        return foo(x + 1);
    }
}

foo(x);
```

Environment Model

Exercise 2.2

```
function alpha(x) {
    let y = 3;

    function beta(x) {
        y = y + x;
        return y;
    }

    return beta;
}

let haha = alpha(5);
haha(1);
```

Overview

Array

Array

- Array is effectively another linear data structure, similar as list.
- Empty array: `[]`
- Array with n element: `[1, 2, ..., n]`
- Access m^{th} element: `arr[m]`
- Array assignment: `arr[m] = "cs"`
- Array length: `array_length(arr)`

Array

Array and list

- List can be implemented using array.
- `pair(a, b)` is just `[a, b]`
- `list(a, b, c, d)` is just `[a, [b, [c, [d, []]]]]`

Array

Is array mutable? - using const

```
const arr = [1, 2, 3];
display(array_length(arr));

// Will this arise an error?
arr[1] = 3;

// Will this arise an error?
arr = [10, 20, 30];

// Will the length of the array change?
arr[4] = 5;
display(array_length(arr));
```

Array

Is array mutable? - using let

```
let arr = [1, 2, 3];

// Will this arise an error?
arr[1] = 3;

// Will this arise an error?
arr = [10, 20, 30];
```

Array

Is array mutable? - conclusion

- What do we really mean by “mutable”?
    - Is the whole data structure mutable?
    - Are the elements of the data structure mutable?
- Array assignment can increase the length of an array.

Array

High-dimensional Array

```
let arr2d = [[1, 2, 3], [4, 5], [6]];

// What's the length of the overall array?
display(array_length(arr2d));

// Does each sub-array have the same length?
display(array_length(arr2d[0]));
display(array_length(arr2d[1]));
```

Array

When to use array

- Implement data structure
- Implement sorting algorithm
- Use together with loop

Your task today ...

- Write a library for array, similar to the one for list.

Loop

while and for loop

- There are two kinds of loops available in Source:

  while and **for**

- They can be converted to each other.

```
for (E1; E2; E3) {
    // ...
}

E1;
while (E2) {
    // ...
    E3;
}
```

Loop

Use loop to “replace” recursion

- With mutable data, we can make use of loop to achieve the same outcome as recursion.

Use while to compute fact(n)

```
let fact = 1;
let k = 1;

while(k < n) {
  fact = fact * k;
  k = k + 1;
}
```

Loop

`continue` and `break`

- `continue`: terminates the current round of the loop and continues the loop with the next round.
- `break`: terminates the current round of the loop and also terminates the entire loop.

Loop & Array

Can you use array and loop to solve these problems?

- Factorial
- Square root
- Power function
- Fibonacci
- Greatest common divisor (GCD)
- Least common multiple (LCM)
- Hanoi tower
- Coin change
- Permutation / combination
- ...

Studio Group Problems

Let's discuss them now.

End

The End

Copyright

Niu Yunpeng © 2017 - 2018. Under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International *License*.

Appropriate credits MUST be given when sharing, copying or redistributing this material in any medium or format. No use for commercial purposes is allowed.

This work is mostly an original by *Niu Yunpeng*. It may either directly or indirectly benefit from the previous work of Martin Henz, Cai Deshun. For illustration purposes, some pictures in the public domain are used. Upon request, detailed acknowledgments will be provided.