

PASS - Computer Systems

Week 11

1. Variable types and scopes

Can you find all static, field, local and parameter variables in the code below?

```
class BankAccount {  
    // Class variables  
    static int nAccounts;  
    static int bankCommission;  
    // account properties  
    field int id;  
    field String owner;  
    field int balance;  
  
    method void transfer(int sum, BankAccount from, Date when) {  
        var int i, j;    // Some local variables  
        var Date due;    // Date is a user-defined type  
        let balance = (balance + sum) - commission(sum * 5);  
        // More code ...  
    }  
}
```

type	variable	declared in	scope	memory segment
Static	nAccounts, bankCommission	class	class	Static
field	id, owner, balance	class	methods can access	this
local	i, j, due	subroutine	subroutine	lcl
parameter	sum, from, when	subroutine parameter list	Subroutine	arg

2. Write a Player class that contains:

- 3 filed variables: **string name, int age, int level**
- 1 static variable: **string color**
- The constructor **new()** takes in 3 parameters: name_, age_. The constructor sets the object's name and age accordingly, and sets level = 0.
- A void function **change_color(string color_)** will set the static variable color to be color_
- A method **upgrade()** increases the level by 1, and return the player's updated level.
- How would you call new(), change_color(), and upgrade()?

```
class Player {
    static string color
    field string name
    field int age
    field int level
    constructor Player new(string name_, int age_) {
        let name = name_
        let age = age_
        let level = 0
        return this
    }
    function void change_colour(string colour_) {
        let color = colour_
        return;
    }
    method int upgrade() {
        let level = level + 1
        return level
    }
}
```

Player.new(x, '0')
do new.change_color(n)
let x = new.upgrade()

3. Tokenizer

JACK program --> tokenizer --> parser --> code generator --> VM code

tokenizer and parser form syntax analyzer

tokenizer, parser and code generator form compiler

Tokenizer: next_token(), have(TokenKind), mustbe(TokenKind)...

- a. Consider these rules for a tokenizer. What tokens would be produced from the following string assuming that whitespace is not returned as tokens?

"23 hobbits50 99chickens"

```
int ::= '0' | (('1'-'9')('0'-'9'))*  
name ::= ('a'-'z') ('a'-'z'|'0'-'9')*
```

Answer:

1. 23 - int
2. hobbits50 - name
3. 99 - int
4. chickens - name

- b. Consider these rules for a tokenizer. What tokens would be produced from the following string assuming that whitespace is not returned as tokens?

"09.06.0.0.0"

```
num ::= ('0' | (('1'-'9')('0'-'9'))*) ('.' ('0'-'9'))*  
dot ::= '.'
```

Answer:

1. 0 - num
2. 9.06 - num
3. "." - dot
4. 0.0 - num
5. "." - dot
6. 0 - num .

4. Why is JACK's grammar LL(1)?

only 1 token is needed to determine the rule.

5. Parser: `parse_class()`, `parse(type)`, `parse`

- a. Consider the grammar rule for a Jack local variable declaration(e.g. `var int x,y;`):

```
varDec ::= 'var' type varname (',' varname)* ';' ;
```

Write example code fragments would be able to parse local variable declarations

6. Construct class-wise and method-wise symbol table for code below:

```
class BankAccount
{
    static int nAccounts ;
    static int bankCommission ;
    field int id ;
    field string owner ;
    field int balance ;
    method void transfer(int sum, BankAccount from, Date whe
n)
    {
        var int i,j ;
        var Date due;

        let balance = (balance + sum) - commission(sum *
5) ;
    }
}
```

Question: how many symbol tables exist in a class? – A class wise symbol table + one symbol table for each subroutine. Commonly implemented as a linked list.

class wise:

name	type	Kind	offset
nAccounts	int	static	0
bankCommission	int	static	1
id	int	this	0
owner	string	this	1
balance	int	this	2

method-wise:

name	type	Kind	offset
sum	int	arg	1
from	BankAccount	arg	2
when	Date	arg	3
Bank Account	this	arg	0
i	int	local	0
j	int	local	1
due	Date	local	2

7. Construct class-wise and method-wise symbol table for code below:

```

1  class Player
2  {
3      field String name;
4      field int age, level;
5      static String color;
6      static int color_intensity;
7
8      constructor Player new(String name_, int age_) {
9          let name = name_;
10         let age = age_;
11         let level = 0;
12         return this;
13     }
14
15     function void change_color(String color_, int intensity) {
16         var int i;
17         let i = intensity;
18         while (i > 0)
19         {
20             let color_intensity = color_intensity * 3;
21             let i = i - 1;
22         }
23         let color = color_;
24         return;
25     }
26
27     method int upgrade() {
28         let level = level + 1;
29         return level ;
30     }
31 }

```

class wise:

name	type	Kind	offset
name	String	field	0
age	int	field	1
level	int	field	2
color	String	static	0
color_intensity	int	static	1

constructor wise:

name	type	Kind	offset
name -	String	arg	0
age -	int	arg	1

function-wise:

name	type	Kind	offset
color_	String	argument	0
intensity_	int	argument	1
i	int	local	0

method-wise:

name	type	Kind	offset
this	Player	argument	0