CS224 – Jack Symbol Table

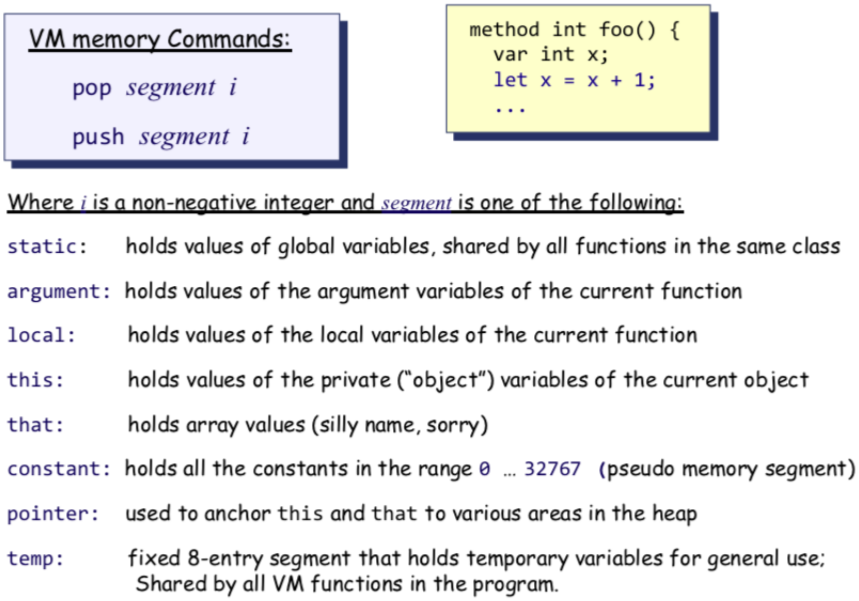
Purpose: We need to create a symbol table for the Jack compiler to hold information about all of our variables.

Knowledge: This activity will help you become familiar with the following content knowledge:

- What information needs to be contained in the Jack symbol table

Activity: With your group perform the following tasks and answer the questions. You will be reporting your answers back to the class in 15 minutes.

1. We will start with reviewing the VM memory commands from CS220:



VM memory Commands:

pop *segment i*

push *segment i*

```
method int foo() {
  var int x;
  let x = x + 1;
  ...
```

Where *i* is a non-negative integer and *segment* is one of the following:

`static:` holds values of global variables, shared by all functions in the same class

`argument:` holds values of the argument variables of the current function

`local:` holds values of the local variables of the current function

`this:` holds values of the private ("object") variables of the current object

`that:` holds array values (silly name, sorry)

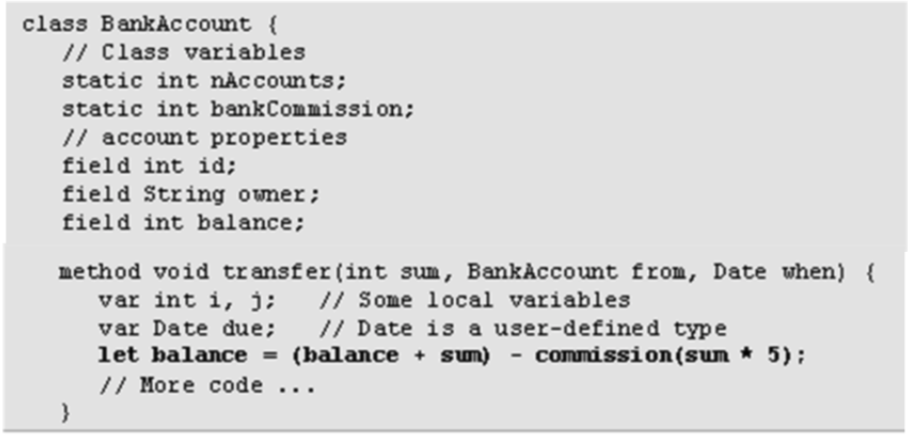
`constant:` holds all the constants in the range `0` ... `32767` (pseudo memory segment)

`pointer:` used to anchor `this` and `that` to various areas in the heap

`temp:` fixed 8-entry segment that holds temporary variables for general use;
Shared by all VM functions in the program.

Give the VM code for the `let` statement in the method `foo` given above.

2. Variable are mapped onto memory segments. Consider the following Jack class:



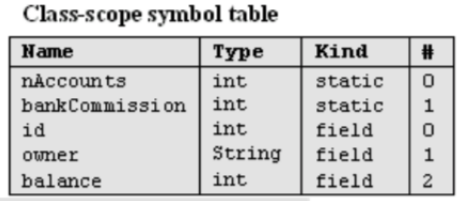
```
class BankAccount {
   // Class variables
   static int nAccounts;
   static int bankCommission;
   // account properties
   field int id;
   field String owner;
   field int balance;

   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 ...
   }
```

   What memory segment and index would be used to access the variable `balance` in the `transfer` method?

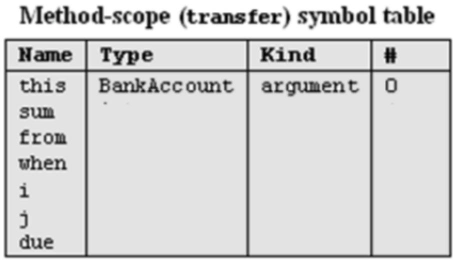
   What memory segment and index would be used to access the variable `sum` in the `transfer` method?

3. We will actually have two symbol tables at any point in time. The first table will be the variables that have class scope for the current class being used. The second table will be the variables that have method scope for the current method being used.
   Here is the class scope table for our BankAccount example. Complete the method scope table for the `transfer` method.



Class-scope symbol table

Name	Type	Kind	#
nAccounts	int	static	0
bankCommission	int	static	1
id	int	field	0
owner	String	field	1
balance	int	field	2



Method-scope (transfer) symbol table

Name	Type	Kind	#
this	BankAccount	argument	0
sum			
from			
when			
i			
j			
due			

   (It looks like you are ready for the Jack symbol table project now!!)