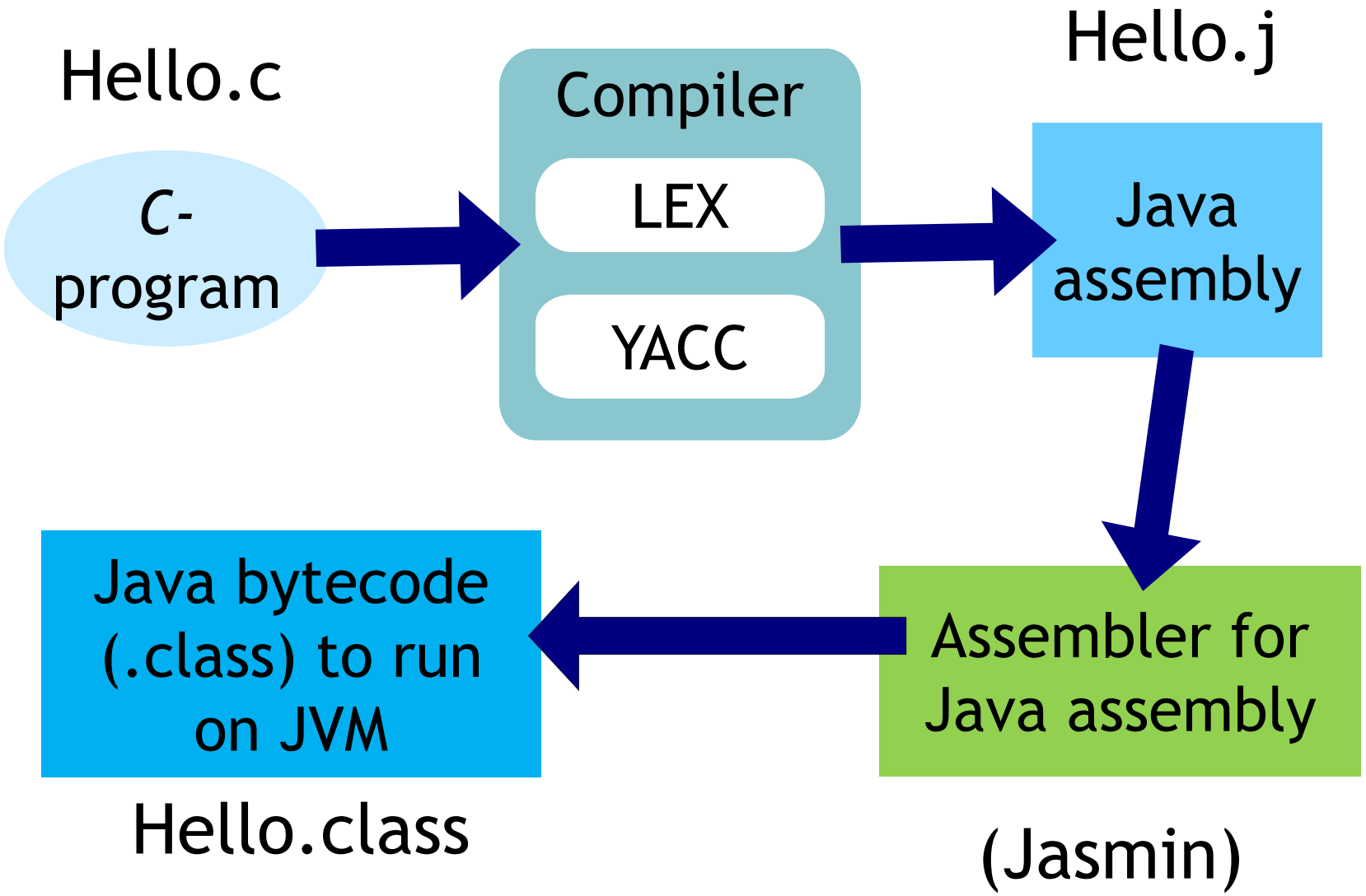
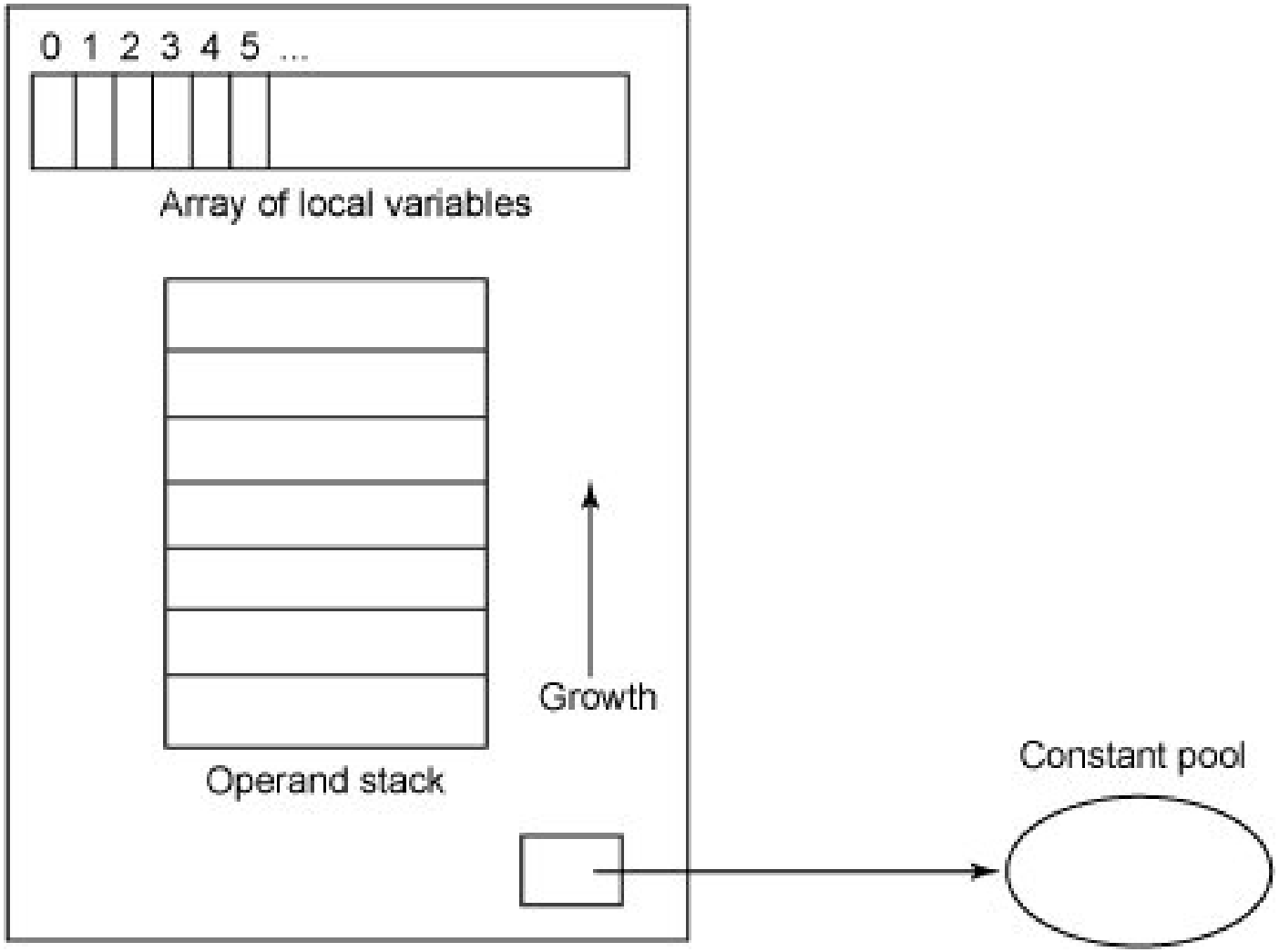
The Goal of Term Project



Java Virtual Machine

- JVM is a stack-based machine
- Each thread has a stack that stores frames
- A frame is created for each method invoked
- Each frame has:
  - An “operand stack” to use to execute instructions in the method, its size should be specified
  - Array of local variables that act as registers
    - Their number should be fixed as the start of the method
  - Reference to the constant pool of the current class
  - Arguments are supplied in the local variables array and their number is given (*this* variable is usually in position 0*)*

JVM Frame


0 1 2 3 4 5 ...
Array of local variables
Operand stack
Growth
Constant pool

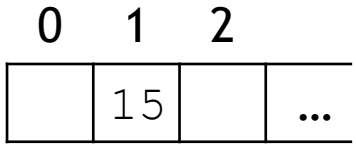
JVM Instruction Set

- A JVM instruction consists of a one-byte opcode specifying the operation to be performed, followed by zero or more operands supplying arguments or data that are used by the operation
- Many instructions have no operands and consist only of an opcode
- Java bytecode

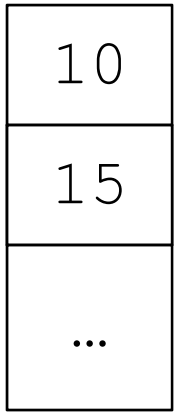
A Simple Example

- A *C*- statement
  - `a = 5 + 10;`
- Java bytecode
  -

```
bipush 5    ;push 5 onto the stack
bipush 10   ;push 10 onto the stack
iadd        ;add top two numbers on
             the stack and leave the
             result on the stack
istore_1    ;pop the result off of the
             stack and store in local
             variable  1
```



Array of local variables



Operand stack

See http://docs.oracle.com/javase/specs/jvms/se5.0/html/VMSpecTOC.doc.html for instruction set summary

Jasmin: Java Bytecode Assembler

- Sun (now Oracle) had not published an assembler format for the Java virtual machine
  - Sun does provide a *javap* program which can print the assembly code in a class file. However, the output of *javap* is inappropriate for use as an assembler format
- Jasmin
  - http://jasmin.sourceforge.net/
  - Provides a Java Assembler Interface
  - A Java bytecode assembler