Jun ho Lim

1. In the ParseInt.j sample Jasmin program given above, decode the nomenclature for type descriptions and use that information to identify the the method signature and return type for each of the method invocations in the ParseInt.j program. Use the table on p205 of Juola to decode the type tags.

For example, in the following type description

```
; ParseInt.j
; Neal Nelson  2008.03.28
; Convert command line argument to integer and print using parseInt.

; class heirarchy
.class public ParseInt
.super java/lang/Object

; standard initializer
.method public <init>()V
  aload_0
  invokespecial java/lang/Object/<init>()V
  return
.end method

.method public static main([Ljava.lang.String;)V
                                    V indicates that the return type is Void
  .limit stack 3

  ; push System.out onto the stack
  getstatic  java/lang/System/out Ljava/io/PrintStream;


  ; push the string in args[0] on the stack
  aload_0         ;load args array address from local #0
  iconst_0        ;load arg array index 0
  aaload          ;load address of string from array of addresses

  ; call Integer.parseInt to convert the string on stack to an integer
  invokestatic java/lang/Integer.parseInt(Ljava/lang/String;)I
                                    V indicates that the return type is Integer

  ; call PrintStream.println()
  invokevirtual java/io/PrintStream/println(I)V
                        (I)V indicates that the return types are Integer and
Void
  return
.end method
```

2. How are strings stored on the JVM stack? Examine the Jasmin program on page 67 of the Juola textbook or the HelloWorld.j program listed above. In particular, look at the ldc instruction that loads a string on the stack. Look up the ldc instruction in the Juola Apx B.
   - Load a single-word entry from the constant pool and push in to the top of the stack.

Jun ho Lim

3. Study the EchoArgs.java program so you understand how it works. Command line arguments in Java (and all other languages running on Unix operating systems) are passed to your main program from the operating system as an array of strings called the *args array*, or just the argument array. Explain how this string array is stored in Java.

   - store command line arguments in an array starting with slot 0 and print out the args[0]. Repeat steps till args.length amount.

4. What does the EchoOneArg.j program do?
   - Load the array and put array position index 0. Store input data in array 0 and print out the args[0] string data.

5. How are arrays stored on the JVM stack? Look up the aload instruction in the Juola Apx B.
   - Pops an integer and an array of addresses (object references) from the stack, then retrieves a value from that location in the one-dimensional array of addresses. The value retrieved is pushed on the top of the stack.

6. How are arrays and strings stored in local variables?
   - The local variables are assigned in to a ram position and then the value is stored in ram memory.

7. What does the aaload jasmin instruction do and what arguments does it expect on the stack?
   - It looks at index int, goes to address of that local variable, then loads that value onto the stack. It expects a value, not an empty location.

8. What does the first "a" in aaload refer to? What does the second "a" in the aaload refer to? Remember, the letter indicates a type.
   - First a is address Second a is Array

9. There are three JVM instructions needed to load an argument array string parameter onto the JVM stack. Explain what each of the three instructions does and how that relates to loading the command line argument parameter.

```
aload_0          ;load args array address from local #0
iconst_0         ;arg array index (args[0])
aaload          ;pushes the value on top of the stack.
```

10. What specifically is on the top of the stack (and in what order) just before the invocation of of println when the EchoOneArg method is called from the commmand line as follows: java EchoOneArg hi ?

```
Swap
getstatic  java/lang/System/out Ljava/io/PrintStream;
aaload
iconst_0
aload_0
```

11. The Juola textbook indicates in the sidebar on p75 that the local variable #0 usually has the object reference. Explain why the EchoOneArg.j and ParseInt.j programs have an array address in local #0. (Hint, read the sidebar on pa75 closely).

    - Because both these programs are calling on a static method, which does not have an object to store in local variable #0 therefore the local variable #0 is available to store a variable.

12. What does the ParseInt.j jasmin program do?
    - It takes an argument from the command line and prints it to the console.

13. What do you need to add to the ParseInt.j jasmin program to input two integers from the command line?
    - Call the PrintStream.println() again
      repeat the same code and push it to the stack again. Take another arugment from the console, and print it out again.

14. Write a jasmin program to input two command line integers, average them, and print the result on the command line. You don't need to run this program for these exercises (you'll do that for the lab).

```
.class public Lab5_AverageLoop
.super java/lang/Object

.method public <init>()V
    aload_0
    invokespecial java/lang/Object/<init>()V
    return
.end method

.method public static main([Ljava/lang/String;)V
    .limit stack 10
    .limit locals 5
    iconst_0
    istore_3
    aload_0
    arraylength
```

```
    istore_1
start:
    iload_1
    ifgt loop
    goto end
loop:
    iload_1
    iconst_1
    isub
    istore_1
    aload_0
    iload_1
    aaload
    invokestatic java/lang/Integer.parseInt(Ljava/lang/String;)I
    iload_3
    iadd
    istore_3
    goto start
end:
    getstatic java/lang/System/out Ljava/io/PrintStream;
    iload_3
    aload_0
    arraylength
    idiv
    invokevirtual java/io/PrintStream/println(I)V

    return
.end method
```