Homework 2

Programming Languages Principles and Implementation

Honglin Yi

October 2017

Question 1: History of programming languages

Put the following programming languages on a chronological timeline. The year must be provided. In addition, indicate the name of the designer of the programming language, where it was created (company, national lab, higher education institution etc.), and the country.

Language	Year	Designer	Places of Invention	Country
Fortran	1957	John W. Backus	IBM	The U.S.
Lisp	1958	John McCarthy	MIT	The U.S.
Cobol	1959		The CODASYL Committee	The U.S.
Pascal	1970	Niklaus Wirth, Kathleen Jensen		The U.S.
C	1972	Dennis Ritchie		The U.S.
Prolog	1972	Alain Colmerauer		The U.S.
ADA	1980	Jean Ichbiah	CII Honeywell Bull	The U.S.
C++	1983	Bjarne Stroustrup		The U.S.
SML	1984	Robin Milner		The U.S.
Perl	1987	Larry Wall		The U.S.
Python	1991	Guido van Rossum		Dutch
Java	1995	James Gosling	Sun Microsystems	The U.S.
ISETL	1995	Gary Levin	Clarkson University	The U.S.
Ruby	1995	Yukihiro Matsumoto		Japan

Language	Year	Designer	Places of Invention	Country
Kotlin	2011	JetBrains	JetBrains	The U.S.

Question 2

Consider the following code. Each draw method has a number.

a)Explain polymorphism on the code above.

Polymorphism is the capability of a method to do different things based on the object that it is acting upon. In other words, polymorphism allows you define one interface and have multiple implementations.

As we have seen in the above example that we have defined the method draw() and have the multiple implementations—two in class Circle, one in its sub class ColoredCircle.

Which draw() method will be called is determined at runtime.

b)c is of type Circle and d is of type ColoredCircle. Can we write d = c;? Why?

No, java can't allow one class cast to its sub-class.

c)c is of type Circle and d is of type ColoredCircle. Can we write c = d;? Why? What happens if we execute the code below? What method called draw is called? Why?

Yes, because according to java's Plolymorphism feature, a ColoredCircle object IS-A Circle.

Run the code below,

```
c = d;
 c.draw();
```

the compiler will call draw() method of ColoredCircle class. Because currently the variable c refers to the ColoredCircle object and ColoredCircle class overrides the draw() method of Circle class, the compiler will call the draw of ColoredCircle class.

Question 3:

Install the following Eclipse Bytecode Outline plugin from: http://asm.objectweb.org/eclipse/index.html or from the Eclipse MarketPlace.

a)What Eclipse version are you using?

Oxygen.1a Release (4.7.1a)

b)What Java version are you using?

1.8.0_144

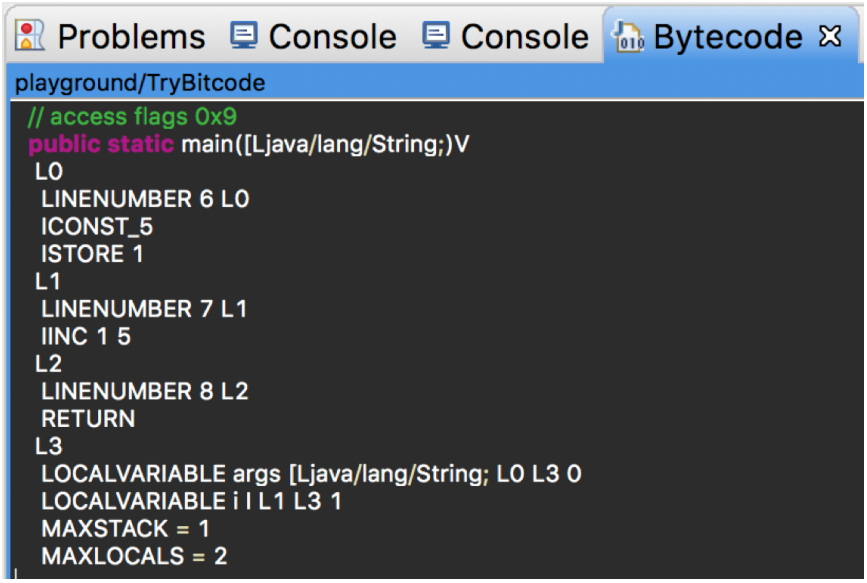
c)What is the Bytecode generated by the following statements?

int i = 5;

i = i+5;

Explain the syntax of the Bytecode. Provide a screenshot to support your work.

ICONST_5
ISTORE 1
IINC 1 5



Each instruction consists of a one-byte opcode followed by zero or more operands. Like in "IINC 1 5", "IINC" is the opcode and there're two operands, variable 5 and constant 5.

d) Compare the Bytecode generated by the 2 functions below and write down your conclusions.

Provide screenshots to support your work.

```
public static int sum_for(int n) {
	int i = 0, sum = 0;
	for (i = 0; i <= n; i++) {
		sum += i;
	}
	return sum;
}

public static int sum_while(int n) {
	int i = 0, sum = 0;
	while (i <= n) {
		sum += i;
		i++;
	}
	return sum;
}
```

```
public static sum_for(I)I
 L0
  LINENUMBER 11 L0
  ICONST_0
  ISTORE 1
 L1
  ICONST_0
  ISTORE 2
 L2
  LINENUMBER 12 L2
  ICONST_0
  ISTORE 1
  GOTO L3
 L4
  LINENUMBER 13 L4
 FRAME APPEND [I I]
  ILOAD 2
  ILOAD 1
  IADD
  ISTORE 2
 L5
  LINENUMBER 12 L5
  IINC 1 1
 L3
 FRAME SAME
  ILOAD 1
  ILOAD 0
  IF_ICMPLE L4
 L6
  LINENUMBER 15 L6
  ILOAD 2
  IRETURN
 L7
  LOCALVARIABLE n I L0 L7 0
  LOCALVARIABLE i I L1 L7 1
  LOCALVARIABLE sum I L2 L7 2
  MAXSTACK = 2
  MAXLOCALS = 3
```

```
public static sum_while(I)I
 L0
  LINENUMBER 19 L0
  ICONST_0
  ISTORE 1
 L1
  ICONST_0
  ISTORE 2
 L2
  LINENUMBER 20 L2
  GOTO L3
 L4
  LINENUMBER 21 L4
 FRAME APPEND [I I]
  ILOAD 2
  ILOAD 1
  IADD
  ISTORE 2
 L5
  LINENUMBER 22 L5
  IINC 1 1
 L3
  LINENUMBER 20 L3
 FRAME SAME
  ILOAD 1
  ILOAD 0
  IF_ICMPLE L4
 L6
  LINENUMBER 24 L6
  ILOAD 2
  IRETURN
 L7
  LOCALVARIABLE n I L0 L7 0
  LOCALVARIABLE i I L1 L7 1
  LOCALVARIABLE sum I L2 L7 2
  MAXSTACK = 2
  MAXLOCALS = 3
}
```

Conclusion: While Loop is more efficient than For Loop, because it use less byte codes.

e) Write the factorial function (with the profile: public static fact(int n)) and describe the bytecode generated by this function.

```
package playground;

public class TryBitcode2 {
    public static int fact(int n) {
        if(n == 0) {
            return 1;
        }
        else {
            return n*fact(n-1);
        }

    }
}
```

```
public static fact(int) : int
 L0
  LINENUMBER 5 L0
  ILOAD 0: n
  IFNE L1
 L2
  LINENUMBER 6 L2
  ICONST_1
  IRETURN
 L1
  LINENUMBER 9 L1
 FRAME SAME
  ILOAD 0: n
  ILOAD 0: n
  ICONST_1
  ISUB
  INVOKESTATIC TryBitcode2.fact (int) : int
  IMUL
  IRETURN
 L3
  LOCALVARIABLE n int L0 L3 0
  MAXSTACK = 3
  MAXLOCALS = 1
}
```

Describe:

JVM loads one int n into the stack, if the value of n is not zero, branch to instruction at branch-offset.

It takes two ints—n & n into the stack, one is for parameter n, one is for local variable n and loads the int value 5 onto the stack.

Then it pops two ints off the operand stack, subtracts the top one from the second (n-1), and pushes the int result back onto the stack.

Then it invoke the static method—fact(int n) and puts the result on the stack and multiplies local variable n and the result. At last, return the int.

f) Choose a tail recursive function and describe the bytecode generated by this function. Compare with the code generated for a recursive function obtained in c).

```
public static int tailfact(int n, int x) {
    if (n == 0) {
        return x;
    }else {
        return tailfact(n-1, n*x);
    }
}
```

```
public static tailfact(int, int) : int
 L0
  LINENUMBER 15 L0
  ILOAD 0: n
  IFNE L1
 L2
  LINENUMBER 16 L2
  ILOAD 1: x
  IRETURN
 L1
  LINENUMBER 18 L1
 FRAME SAME
  ILOAD 0: n
  ICONST_1
  ISUB
  ILOAD 0: n
  ILOAD 1: x
  IMUL
  INVOKESTATIC TryBitcode2.tailfact (int, int) : int
  IRETURN
 L3
  LOCALVARIABLE n int L0 L3 0
  LOCALVARIABLE x int L0 L3 1
  MAXSTACK = 3
  MAXLOCALS = 2
```

The bytecode is very similar with the bytecode of normal recursive function, except that tail recursive function will load the value of the extra variable x into the stack. It means that tail recursive function will cost more space in the stack.

Question 4:

a. Write a PROLOG program that describes the British family until nowadays. Kate, William and their children should be cited in the facts. Your program will start with the facts available in the slides (slide 31) and ends with Kate, William and their children.

```
male(Edward VII).
male(George V).
male(George VI).
male(Prince Charles).
male(Prince William).
male(Prince George).
female(Diana).
female(Kate).
female(Alexandra).
female(Elizabeth II).

Parent(Edward VII, George V).
parent(Victoria, Edward VII).
parent(Alexandra, George V).
parent(George VI, Elizabeth II).
parent(George V, George VI).
parent(Elizabeth II, Prince William).
parent(Prince Charles, Prince William).
parent(Diana, Prince William).
parent(Kate, Prince George).
parent(Prince William, Prince George).
parent(Kate, Princess Charlotte).
parent(Prince William, Princess Charlotte).

father(X,Y) :- male(X),parent(X,Y).
mother(X,Y) :- female(X),parent(X,Y).
son(X,Y) :- male(X),parent(Y,X).
daughter(X,Y) :- female(X),parent(Y,X).
grandfather(X,Y) :- male(X),parent(X,Somebody),parent(Somebody,Y).
```

b. Write a **rule** that describes the father predicate. *Father(X,Y)* means that *X* is the father of *Y*.

```
father(X,Y) :- male(X),parent(X,Y).
```

Question 5:

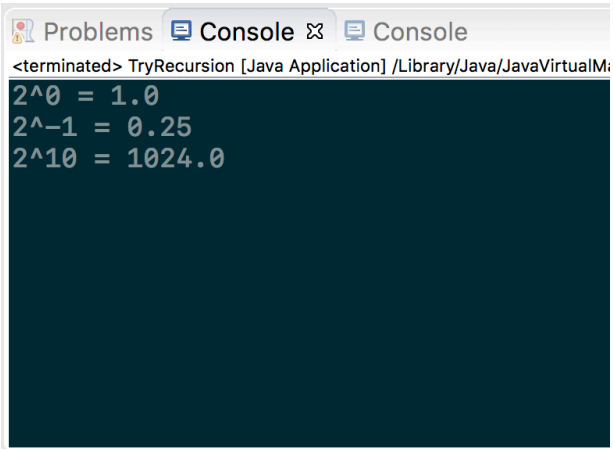
Write a **recursive** function *recPow* that computes 2^n for n >= 0 in Java. The function will have the following profile: public static int recPow(int n)
The function must consider all cases and be tested exhaustively. Show your testing! -1=1/2 0=1 n = (n+1)/2

```
public class TryRecursion {

    public static void main(String[] args) {
        System.out.println("2^0 = "+recPow(0));
        System.out.println("2^-2 = "+recPow(-2));
        System.out.println("2^10 = "+recPow(10));
    }

    public static Double recPow(int n) {
        if (n == 0) {
            return 1.0;
        }else if(n > 0){
            return 2*recPow(n-1);
        }else {
            return recPow(n+1)/2;
        }
    }

}
```



Question 6:

Write a **recursive** function merge that merges 2 arrays in Java. . The function will have the following profile:

public static int[] mergeSort(int[] a, int[] b)

You will use the split function of slide 18 (odd and even positions).

The function must be tested exhaustively. Show your testing!

If you use code online, you will need to cite your sources.

```java
public static int[] mergeSort(int[] a, int[] b) {
    int[] c = sort(a);
    int[] d = sort(b);
    return mergeSortedArray(c, d);
}
```

```java
public static int[] sort(int[] a) {
    if(a.length == 1) {
        return a;
    }
    int L[] = new int [a.length/2];
    int R[] = new int [a.length-L.length];

    System.arraycopy(a, 0, L, 0, L.length);
    System.arraycopy(a, L.length,R, 0, R.length);

    sort(L);
    sort(R);
    return mergeSortedArray(L, R);
}
```

```java
public static int[] mergeSortedArray(int[] arr1, int[] arr2){
   int i = 0, j = 0, k = 0;
   int n1 = arr1.length;
   int n2 = arr2.length;
   int[] arr3 = new int[n1+n2];

   while (i<n1 && j <n2){
      if (arr1[i] < arr2[j])
          arr3[k++] = arr1[i++];
      else
          arr3[k++] = arr2[j++];
      }

   while (i < n1)
      arr3[k++] = arr1[i++];
   while (j < n2)
       arr3[k++] = arr2[j++];
   return arr3;
  }
```