**CS 3204N – Programming Languages**

**Chapter 9 – Subprograms**

1) Define the terms: a) Subprogram definition, b) Subprogram call, c) Subprogram header
2) What is one characteristic of Python functions that sets them apart from the functions of other common programming languages?
3) What are the two ways that a non-method subprogram can gain access to the data that it is to process?
4) Give an advantage and a disadvantage of positional and keyword parameters.
5) Distinguish between a procedure and a function and name a PL that supports procedures.
6) What is a generic subprogram?
7) Discuss two advantages and two disadvantages of stack-dynamic local variables.
8) Differentiate in mode, out mode, and inout mode of parameter passing.
9) a. Discuss the parameter passing method of Python and Ruby.
   b. Discuss how a reference to an array is handled when passed as a parameter in pure object oriented languages.
10) Christine and Pena are arguing about parameter passing in Java. Specifically, Christine told that Java does not accept an integer argument that is passed into a function with a float parameter. Pena on the other hand said that this is not allowed in Python. Who among the two are correct?
11) a. In C, why is it required to indicate the column count of a 2D array as a formal parameter of a subprogram?
    b. Given the prototype `void Regacho(float *Hilvano, int num_rows, int num_cols);` Provide the C statement that can be used to move the value of the variable Lord to the [Christian][Carl] element of the parameter Hilvano in Regacho()? Here, Hilvano is a pointer to a 2D array of float values, num_rows and num_cols represent the array's row and column size respectively.
    c. Mr. Wayne argues that in Java, the formal parameter of a matrix appears with two sets of empty brackets like void check(int matrix[ ][ ]), not needing to specify the number of columns since in a matrix the rows can have different lengths. Mr. Dayata argues that the same is true in Python. Who among the two are correct?
12) Can we construct an equivalent swap function in Java that is similar to the code fragment in C? Explain.
13) a. Differentiate shallow binding, deep binding, and ad-hoc binding.
    b. Consider the execution of sub2 when it is called in sub4. Determine the referencing environment and that execution and the output for (i) shallow binding, (ii) deep binding, and (iii) ad hoc binding.

```
void swap2(int *a, int *b) {
  int temp = *a;
  *a = *b;
  *b = temp;
}
```

#12

#13b

```
function sub1() {
  var x;
  function sub2() {
    alert(x); // Creates
    };
  function  sub3() {
    var  x;
    x = 3;
    sub4(sub2);
    };
  function  sub4(subx) {
    var  x;
    x = 4;
    subx();
    };
  x = 1;
  sub3();
};
```

14) a. Name a PL where functions cannot have side effects.
    b. How can multiple values be returned at a time in (i) C, (ii), Java, (iii) Python?
15) Provide the outputs (or explain the errors) in each of the following scenarios:

```
class Q15_A{
    public float test (float x){
        return x+x;
    }
    public float test (int y){
        return y-y;
    }
}

class HelloWorld {
    public static void main(String[] args) {
        int y = 5;
        float ans;
        Q15_A a = new Q15_A();
        ans = a.test(y);
        System.out.println(ans);
    }
}
```

```
class Q15_B{
    public float test (double x){
        return x+x;
    }
    public float test (int y){
        return y-y;
    }
}

class HelloWorld {
    public static void main(String[] args) {
        int y = 5;
        float ans;
        Q15_B a = new Q15_B();
        ans = a.test(y);
        System.out.println(ans);
    }
}
```

```
class Q15_C{
    public int test (int x){
        return x+x;
    }
    public float test (int y){
        return y-y;
    }
}

class HelloWorld {
    public static void main(String[] args) {
        int y = 5;
        int ans;
        Q15_C a = new Q15_C();
        ans = a.test(y);
        System.out.println(ans);
    }
}
```

16) Give a scenario where overloaded subprograms can lead to ambiguous subprogram calls.
17) Explain subtype polymorphism and parametric polymorphism.
18) Which of the following PLs support user-defined operator overloading? [ C, C++, Java, Python ]
19) What is a closure and what are the language characteristics that make closures useful?
20) What is a coroutine and what are its differences from conventional subprograms?

## Chapter 10 – Implementing Subprograms

**#24:**

```
#define nathaniel OXF
void hannah(float ruth[], int labana){
   int marc[nathaniel];
   float valeros;
};
```

21) Enumerate the actions required in the semantics of a:
   a. Call to a “simple” subprogram
   b. Return from a “simple" subprogram

22) List the five items the call and return actions for “simple” subprogram require storage for.

23) a. What is an activation record?
   b. What are the contents of a typical activation record for a language with stack-dynamic local variables?

24) How many elements are there in the activation record for function hannah() ?

**#27:**

```
function J(gwapos){
  var x=1;
  function jade(){
    var a=2, b=3, c=4;
    function jogie(){
      var a=5, d=6;
      a = b + c; // HERE
      console.log(x);
      console.log(a);
    }
    function jomar(x){
      var b=7, e=8;
      function jstewart(){
        var c=9, e=10;
        jogie();
        e = b + a;
      }
      jstewart();
      a = d + e;
    }
    jomar(7);
  }
  jade();
}
```

25) Differentiate between a static link and a dynamic link in terms of their usage.

26) Discuss the two-step access process required in referencing to a nonlocal variable in a static-scoped language with nested subprograms.

27) In the JS code snippet at the right, function J() is called with an argument of 69.
   a) Show the stack situation when execution first arrives // HERE. Include the dynamic and static chains.
   b) Determine the output of the code snippet.
   c) In // HERE, what are the chain_offset/local_offset pair values of the references to (i) a, (ii) b, and (iii) c ?

**#29-30:**

```
void fun1() {
  float a;
  . . .
}

void fun2() {
  int b, c;
  . . .
}

void fun3() {
  float d;
  . . . <--------- 1
}

void main() {
  char e, f, g;
   . . .
}
```

28) List and differentiate two distinct ways in which local variables and nonlocal references to them can be implemented in a dynamic-scoped language.

29) Show the stack with all activation record instances, including static and dynamic chains, when execution reaches position 1 in the following skeletal program. This program uses the deep-access method to implement dynamic scoping.

   (Calling sequence: main calls fun2, fun2 calls fun1, fun1 calls fun1, fun1 calls fun3)

30) Assume that the program of Problem 27 is implemented using the shallow-access method using a stack for each variable name. Show the stacks for the time of fun3()’s execution.

## Chapter 12 – Support for Object-Oriented Programming

31) a. What were the first languages that supported the concept of object-oriented programming?
   b. A language that is object oriented must provide support for three key language features. List them.

32) What are the main issues in programming with abstract data types? Cite two.

33) a. In the context of OOP, what are messages and message protocols?
   b. Differentiate passing a message from calling a subprogram.
   c. Differentiate abstraction from abstract classes.

34) a. Why does Java not support multiple inheritance?
   b. What does Java implement as a substitute of multiple inheritance?

35) Explain the advantages and disadvantages of:
   a. Implementing an object system that subsumes all other concepts of type (like scalar variables)
   b. Restricting all objects to be heap-dynamic
   c. Making deallocation of heap-dynamic objects explicit
   d. Implementing nested classes

36) Explain the following concepts or keywords in Java:
   a. boxing  b. mixing inheritance  c. @override  d. Vector  e. final (in classes and methods)

37) The final “standing question” in the PL class was about garbage collection in Java. It was mentioned that this is done implicitly to help JVMs in-memory optimization.
   a. What is the specific name of the method, and from what class is this method found?
   b. Give three scenarios where the said method can be triggered prior to the completion of the program execution. Support each scenario with an example code fragment.
   c. Can the said method be explicitly called? If yes, provide the statement. Otherwise, explain why.

38) Can a constructor in Java be: (a) overloaded, (b) overridden, (c) static, (d) final, (e) abstract, (f) private?

39) Differentiate between a class instance record (CIR) and a virtual method table (vtable).

40) a. Discuss the terms reflection, metadata, and introspection.
   b. Give five program elements that can be accessed with reflection in both Java and C#.
   c. Discuss the uses and drawbacks of reflections.
   d. Discuss the following reflection-supporting methods in Python: dir, type, isInstance, callable