

# CPTS 355 Sample Final Exam

NAME \_\_\_\_\_

**Directions:** This exam has 8 pages (including this one) printed double-sided. None should be blank. Check now to make sure your exam is complete. Answer/solve all the questions/problems. If a question asks “why”, asks you to “explain”, or asks you to “describe” then you must write in complete sentences.

The exam is open-book, open-notes; you may **not** use a computer, PDA, cell phone, etc. during the test.

1a) [5] Recall that ML uses static scoping. What is the value of the following ML expression? Explain how you determined your answer.

```
let
  val a = 4
  fun b () = a
  fun c () = let
    val a = 6
  in
    b()
  end
in
  c ()
end
```

1b) [5] *What if* ML used dynamic scoping. What is the value of the expression given in 1a) in that case? Explain how you determined your answer.

(2) [10] What is the type of the following ML expression?

```
fun ok t b [] = b
  | ok t b (x::xs) = if (t x) then (ok t b xs) else false
```

3) [25] Identify each of the following C++ examples as exhibiting parametric polymorphism, inheritance polymorphism or ad-hoc polymorphism.

```
a) class A {
    public:
        int x;
        virtual int getx() {return x;}
        A() {x = 3;}
};

class B: public A {
    public :
        int x;
        B() { x = 5; }
        virtual int getx() { return x; }
};

A *a = new B();
```

```
b) class C {
    private:
        int x;
    public:
        void M() {x = 0;}
        void M(int y) {x = y;}
};
```

```
c) template<typename T> class P {
    T x;
    public:
        bool M(int y) {return true;}
};
```

d) What is the value of `a->getx()` in example a) above? Explain.

e) What is the value of `a->x` in example a) above? Explain.

4) [10] Consider the following python code fragment:

```
def r (x):
    def q (x):
        try:
            return p()
        except "oops", v:
            return v+3
    def p ():
        if x==0: raise oops,7
        else: return 5
    try:
        return q(x-4)
    except "oops", v:
        return v-3
```

What value is returned by `r (4)`? Why?

What value is returned by `r (0)`? Why?

5) [20] (**two parts**) Multiple inheritance poses a number of problems for language designers and implementers. Write separate answers for the following problems.

**a)** Describe what leads to a *name clash*. Illustrate your answer with example **C++** code. What are some strategies adopted in different languages for dealing with name clashes?

**b)** Describe *diamond inheritance*. Illustrate your answer with **Python** code exhibiting diamond inheritance. What program design issues are associated with diamond inheritance?

6) [20] **Draw the call stack** at the point marked by ← in the following Python program. Clearly label each activation record with the name of the procedure that it represents and show the static and dynamic links in each activation record. (Recall that in Python assigning to a variable creates an instance of that variable in the scope where the assignment occurs; so in this program there are separate variables named y associated with procedures main, A, and the second B, and a separate variables x associated associated with main and the first B.)

```
def main ():
    y = 5
    x = 12
    def C ():
        def B ():                # this is "the first B"
            x = 14
            B()
            z = y+x ←
            print z
    def A ():
        y = 11
        def B ():                # this is "the second B"
            y = 7
            C()
    A()
```

Using your diagram of the call stack, explain how the uses of variables x and y at the ← are resolved to the correct binding occurrences (either the one in A, the one in B, or the one in Main).

7) Consider the following code in the language C-by-value-result which is like C but parameters are passed using the **value-result** method.

```
int x = 4;
void p(int y, int z) {
    y = z*3;
    z = z-2;
}
p(x, x);
```

7a) [5] What are the possible values of variable x at the end of execution? Describe the execution of the program that leads to each possible result.

7b) [5] If the parameter passing rule was pass-by-reference in the above code, what would be the value of x at the end of execution? Explain the execution of the program that leads to this result.

8) [10] Explain in detail why reference counting garbage collectors do not work when unreachable data structures contain cycles.

9) [20] Given the constraints  $X$  is in the set  $\{5,7,11,13, 17, 23\}$  and  $Y$  is in the set  $\{2,3,5,7,11,13\}$  and  $X+Y=28$ :

9a) What is (are) the basic constraint(s) of this problem? What is (are) the non-basic constraint(s)?

9b) What is (are) the initial content(s) of the constraint store? What is (are) the initial propagator(s)?

9c) Show step by step how constraint propagation narrows the domain for  $X$  to  $\{17,23\}$  and for  $Y$  to  $\{5,7,11\}$ .

9d) Show step by step how propagation following distribution on  $X=17$  then leads to the answers.

10) [20] In reference-counting garbage collection a fundamental operation is incrementing the reference count of an object:

$$rc = rc + 1$$

10a) How would this be implemented as machine instructions?

10b) How many different states occur during the execution of the code sequence you provided in 10a ?

10c) If one thread is performing a reference count increment on an object and another thread is concurrently performing a decrement, what are the possible final values of the reference count. Assume the reference count was 1 before either thread began its operation.

10d) What mechanism could be used to prevent the possibility of the wrong final reference counts in 10c ?