# COP4020 Fall 2001 – Final Exam

Name: _____ (Please print)
*Put the answers on these sheets. Use additional sheets when necessary. Show how you derived your answer (this is required for full credit and helpful for partial credit). You can collect 100 points in total for this exam.*

1. What is a *statically allocated* variable? (mark **one**) (4 points)

   (a) a variable allocated at an absolute address in a program's data space
   (b) a variable allocated on the stack
   (c) a variable allocated on the heap
   (d) a variable with limited visibility

2. What kind of programming construct is the following C construct? (mark **one**) (4 points)

   ```
   switch (val) { case N: ... case M: ... }
   ```

   (a) a sequencing statement
   (b) a selection statement
   (c) an iteration statement
   (d) a nondeterministic statement

3. *Non-structured* control flow means that ... (mark **one**) (4 points)

   (a) ... C `struct`s are not used in a program
   (b) ... `goto`s are used
   (c) ... proper indentation is not used
   (d) ... concurrency is not used

4. Which of the four sentences below is *false*? (mark **one or <u>more</u>**) (4 points)

   (a) in C++ the binding time of a variable to its type declaration is at run time
   (b) logically controlled *pretest loops* check loop conditions before each iteration
   (c) in a *statically scoped language* the exact storage location of a variable can always be determined at compile time
   (d) an *l-value* is a logical value

5. Suppose a programming language uses garbage collection. What kind of (de)allocation problems do *not* occur? (mark **one or <u>more</u>**) (4 points)

   (a) dangling references
   (b) dereferencing uninitialized pointers
   (c) memory leaks
   (d) internal and external heap fragmentation

6. What is *short-circuit evaluation* of Boolean expressions? (mark **one**) (4 points)

   (a) Boolean expressions are evaluated at compile time
   (b) the logical result of a Boolean operator always evaluates to the same value at run time
   (c) the evaluation of an operand of a Boolean operator is skipped when the logical result is determined from the evaluation of another operand
   (d) if both operands of a Boolean operator are the same, then only one operand will be evaluated

7. The name of the notation used for Scheme's operators and function calls is ... (mark **one**) (4 points)

   (a) prefix notation
   (b) postfix notation
   (c) infix notation
   (d) Cambridge Polish notation

8. Which of the following allocation methods is the most time-efficient, i.e. is the fastest? (mark **one**) (4 points)

   (a) static allocation
   (b) stack allocation
   (c) heap allocation

9. Some languages support keyword parameters (a.k.a. named parameters). What are these? (mark **one**) (4 points)

   (a) a qualifing keyword is used to specify in/out parameter passing modes
   (b) the name of the formal parameter can be used to bind the formal parameter to the actual parameter in a subroutine call
   (c) these parameters are passed by name
   (d) the name of a formal parameters can only be choosen from a fixed set of parameter names

10. Mark the entries in the table indicating which parameter passing modes pass parameters in and/or out (use **yes/no**) (5 points)

| Passing Mode | In | Out |
|---|---|---|
| Value | | |
| Reference | | |
| Sharing | | |
| Value/Result | | |
| Name | | |

11. Draw the layout of a *subroutine frame*. Identify the fields and describe their purpose. (5 points)

12. Describe the *first-fit* and *best-fit* algorithms for heap allocation. (5 points)

13. What is a *subroutine closure*? (5 points)

14. Explain the difference between the *value model of variables* and the *reference model of variables*. (5 points)

15. Consider the following Pascal program:

```
  procedure P1(A1 :  integer)
    var X : integer;
    var Y : integer;
    procedure P2(A2 :  integer)
      var X : integer;
      procedure P3(A3 :  integer)
      begin (* P3 *)
        ...  ⇐
      end (* P3 *)
    begin (* P2 *)
      P3(X); (* call P3 *)
    end (* P2 *)
  begin (* P1 *)
    P2(X); (* call P2 *)
  end (* P1 *)
begin (* main program *)
  P1(0); (* call P1 *)
end
```

(a) What is the *reference environment* at the location indicated by ⇐, i.e. which variables and arguments are visible at this location? (4 points) (Answer in the form of: `A3` of `P3`, etc.)

(b) The main program calls `P1`, and `P1` calls `P2`, and `P2` in turn calls `P3`. What is the stack layout after these calls? Show the stack with subroutine frames and static links. (4 points)

16. Consider the following Pseudo-code program:

```
x :  integer /* global declaration with initial value */
procedure inc(y :  integer)
  x := 1
  y := y+x
begin /* main program */
  x := -1
  inc(x)
  write_integer(x)
end
```

(a) Show the value printed for each of the parameter passing modes in the table below. (8 points)

| | By value | By reference | By value/result | By name |
|---|---|---|---|---|
| Output: | | | | |

(b) Assume parameter passing by reference is used. What are the *aliases* in the program? (4 points)

17. Consider the following Pseudo-code program:

```
x :  integer := 1 /* global declaration with initial value */
y :  integer := 2 /* global declaration with initial value */
procedure add
  x := x+y
procedure do(P : procedure)
  x :  integer /* local declaration */
  x := 2
  P()
begin /* main program */
  y :  integer /* local declaration */
  y := 3
  do(add)
  write_integer(x)
end
```

(a) What is the value printed in case the language uses *static scoping* (i.e. like in C/C++)? (3 points)

(b) What is the value printed in case the language uses *dynamic scoping* with *deep binding*? (3 points)

(c) What is the value printed in case the language uses *dynamic scoping* with *shallow binding*? (3 points)