CS7: Exam 3

TF: H. Chad Lane
hcl@cs.pitt.edu

NAME:

DO NOT TURN TO THE NEXT PAGE UNTIL I ANNOUNCE IT!

Read all instructions carefully!

- **If you would like your final exam score and course grade posted on the web, enter a 5 character (combo of letters and numbers preferred) at the bottom of this page.**
- You may not leave the room for any non-life-threatening reason. **IF YOU NEED TO USE THE RESTROOM, GO NOW!** before reading on.
- This is a closed book and closed notes exam. You may not consult with anyone but me during the 2 hour test period. Nothing should be in view except this exam and a pencil or pen (i.e., keep all books, bags, notebooks, etc. under your chair; out of the sight of *anyone*).
- I will not give hints, but will clarify a question if you can convince me it is poorly phrased. If you think this is the case for a particular question, find me in the room and come and ask me. Walk with your exam concealed.
- There are 200 points total on **12 pages**, with no bonus.
- Answer all questions to the best of your ability. Partial credit will be given, so do your best on each. If you are unable to answer a question completely for some reason, try to explain in words what needs to be done.
- Assume all code compiles and runs correctly, unless stated otherwise.
- The last sheet is almost entirely blank. Use this for your scratch work. If you have a final answer written somewhere besides the space provided for each question, **you must indicate, at the location of the original question, where I can find your answer**.
- I'll give 10 and 5 minute warnings.
- Read and sign the statement on the last page before handing in your exam.
- Have a great fall!

5 CHAR CODE:

(only if you want your grade posted on the web)

Non-Java Questions

1. (20 pts; @4 ea.) Circle the best answer for each and make sure and read all possible answers before you circle one.

   (a) Subrange types are...

      i. always ordinal.
      ii. allowed as array index types.
      iii. allowed as function return types.
      iv. declared in the type section of a program.
      v. all of the above.

   (b) Suppose the compiler tells you about an error and does not create an executable file. Which is the most likely reason?

      i. You had a syntax error.
      ii. You forgot to include enough comments explaining your code.
      iii. You had a logical error.
      iv. You had a run-time error.
      v. all of these things could prevent an executable file from being created.

   (c) Suppose you needed to sort 1000 integers. Which data representation would be best suited for the task?

      i. 1000 integer variables
      ii. a 2-dimensional array of integers
      iii. a record
      iv. a file
      v. an array

   (d) Which statement about records is correct?

      i. Records must have at least 2 fields.
      ii. The fields of a record can be of different types.
      iii. Records and arrays are interchangeable.
      iv. You can use records without defining them first in the type section.
      v. none of these are correct.

   (e) Suppose you have an array containing `(12, 14, 8, 25, 7)`. What would it look like after *two* iterations of a selection sort?

      i. `(12, 25, 8, 14, 7)`
      ii. `(7, 8, 12, 14, 25)`
      iii. `(25, 14, 12, 7, 8)`
      iv. `(7, 8, 14, 25, 12)`
      v. `(25, 12, 14, 8, 7)`

2. (9 pts; @3 ea.) Each of the following tasks are suitable for a *single* subprogram. For each, indicate if a **function** or **procedure** is best and briefly state why. **DO NOT WRITE ANY PASCAL CODE!**

   (a) Find the index of the smallest element in an array of integers (assuming no duplicates).

   (b) Given some real number as input, find a numerator and denominator that would be equivalent. For example, for an input of 0.75, the answer would be 3 and 4.

   (c) Determine if the values of each element in one array of integers are all less than the corresponding elements of another array (of the same type).

3. (10 pts) Write a Pascal procedure that takes in two integers and swaps them if the first is bigger than or equal to the second. Otherwise, it leaves them alone.

4. (18 pts) Read the following problem statement then describe how you would plan a solution. Indicate clearly how you would break the problem up into manageable pieces, your main programming goals, the key data, and give an idea of how top-level control would work (i.e., main block layout). Remember, you don't have to solve all (or even many) of the details here.

   Bowling. Write a program that keeps track of 1 to 4 bowlers and allows them to input their scores frame by frame. Along the way, the program should display the current score and who's turn it is. After 10 frames (a complete game), it should display the final scores (and names) from best to worst and give them the option to play again.

   A *frame* in bowling is represented by two numbers, pins hit in the first throw and pins hit in the second. If all 10 are gone in the first, that is called a *strike*. If first ball is not 10, but the sum of the two numbers is 10, then that is called a *spare*. In all other cases (i.e., the sum is < 10), is a called an *open* frame.

   The score accumulates along the way to the 10th and final frame. There are special rules for updating scores if the player gets strikes or spares (but don't worry about them here). Just be aware that it isn't the simple sum of 10 numbers. The 10th frame consists of 3 throws, but the third is only allowed if the first two are strikes or a spare.

5. (16 pts; @8 ea.) What is the output of the following two code segments (assume all variables have been declared and are integers)?

```
sum := 100;
for j := 2 to 6 do
  begin
    writeln(sum);
    sum := sum - j;
  end;
```

```
k := 3;
repeat
  if (k mod 2 = 1) then
    k := 1 + k*3
  else
    k := k div 2;

  writeln(k);
until ( k < 4 );
```

6. (16 pts; @2 ea.) Circle **T**(true) or **F**(false) for each of the following:

   (a) **T** / **F** Global variables are always accessible from anywhere in a program.

   (b) **T** / **F** There is no restriction on the element type of an array.

   (c) **T** / **F** There is no restriction on the index type of an array.

   (d) **T** / **F** `repeat-until` is a *pre-test* loop.

   (e) **T** / **F** Type checking of an assignment statement occurs at compile-time.

   (f) **T** / **F** Using value parameters if more efficient (than var paramters) because they do not require the allocation of a new memory location to hold the incoming value.

   (g) **T** / **F** A subprogram parameter can be a programmer-defined type.

   (h) **T** / **F** The following is a valid boolean expression that determines if `n` (an integer variable) is between 1 and 100:

   ```
   (1 <= n <= 100)
   ```

7. (16 pts; @2 ea.) Given the following declarations:

```
type
  BIArray = array[boolean] of integer;
  IRArray = array[1..100] of real;
var
  A1 : BIArray;
  A2 : IRArray;
  i : integer;
  x : real;
  b : boolean;
```

Indicate if the following statements are **V**alid or **I**nvalid. Assume all data items contain meaningful values. You do **not** need to explain your answers unless you feel a strong need to.

(a) **V** / **I** `b := i < 0;`

(b) **V** / **I** `b := A1[true] in [5, 10, 15, 20];`

(c) **V** / **I** `i := x[1];`

(d) **V** / **I** `x := A1[not b];`

(e) **V** / **I** `readln(A1[b]);`

(f) **V** / **I** `A2[x] := 4.5;`

(g) **V** / **I** `writeln(i + b[true]);`

(h) **V** / **I** `x := A2[(b mod 100) + 1];`

8. (10 pts) Write a record declaration for a building. It should contain fields for the name, number of floors, and square footage. After that, write a declaration for an array of 100 of these.

9. (15 pts) Given the 3 by 4 array `A` (of integers) below, draw a similar picture indicating what `A` will look like *after* the code segment.

```
          1    2    3    4
        --------------------
        | 2  | 0 |  7 |  2 |
        --------------------
A =     | 4  | 4 |  2 | 10 |
        --------------------
        | 12 | 1 | 11 |  1 |
        --------------------
```

```
for i := 1 to 3 do
  for j := 1 to 4 do
    begin
      if ( A[i,1] < 10 ) then
        A[i,1] := A[i,1] + A[i,j]
      else
        A[i,j] := 0;
    end;
```

10. (30 pts; @ 10 ea.) Given the following declarations:

```
type
  Point = record
            x, y : Integer;
          end;
  PointArray = array[1..10] of Point;
var
  Path : PointArray;
```

`Path` is an array of 10 `Point`s. You may assume each point in the array has been initialized to (0,0) – that is, the field `x` and `y` for each element in the array equals 0.

(a) Write a procedure that takes in a *PointArray* as a parameter and reads in values for the whole thing (i.e., all 10 locations – you should use a loop!).

(b) Draw a nice picture of `Path` (note: draw a picture of the *data structure*, not of some path it might represent). Just put in any values you want for the x and y coordinates of each element - you are drawing an array of records.

(c) The idea here is that each location in Path represents a coordinate visited by some traveler. So, if s/he started at the origin and then went up one, `Path[0]` would hold (0,0) and `Path[1]` would hold (0,1), and so on.

For this part of the problem, write a code segment (no subprogram) that works through `Path` and prints out "JUMP!" if either the x or y field increases by more than two from the previous location. HINT: you might make your loop go from 2 to 10.

11. (10 pts) What is the output of the following program?

```
program rockandroll;

var
   a, b : integer;

procedure ren(x : integer);
begin
  a := a + x;
end;

procedure stempy(var x : integer);
begin
  x := x - 8;
end;

procedure fry(var b : integer);
begin
  b := 0;
end;

begin
  a := 10;
  b := 20;
  writeln(a:4, b:4);

  ren(5);
  writeln(a:4, b:4);

  stempy(b);
  writeln(a:4, b:4);

  fry(a);
  writeln(a:4, b:4);

  stempy(b);
  ren(b + 2);
  writeln(a:4, b:4);
end.
```

Java Questions

12. (10 points) Explain how it is that when you compile a java program into a byte-code file (i.e., a `.class` file), that this is runnable an pretty much any modern computer anywhere. Also explain why this is *not* the case for Pascal.

13. (10 points; @2 ea.) For each of the following Pascal/Java pairs, indicate if the java statement is a **C**orrect or **I**ncorrect translation of the Pascal statement.

	Pascal	Java
C / I	`writeln('The value is', value);`	`println('The value is' + value);`
C / I	`sum := sum + i; i := i + 1;`	`sum := sum + i++;`
C / I	`if (JBmeasure > 0.0) then`	`if (JBmeasure > 0.0)`
C / I	`if (change mod 25 = 0) then`	`if (change % 25 = 0)`
C / I	`while ((a < 0) and (b <> c)) do`	`while ((a < 0) && (b != c))`

14. (10 points) Suppose you have a java `Stove` class available to you (as in a kitchen stove). A `Stove` object has four burners numbered 1 to 4 and each can be set to heat level which is a number between 0 (off) and 10 (high). The class has the following methods:

    - `void setBurner(int b, int h)` – sets burner number `b` to setting level `h`.
    - `void turnOffBurner(int b)` – sets burner number `b` to 0 heat.
    - `int getBurnerSetting(int b)` – returns the heat setting for burner `b`.

    Assume the constructor starts `Stove` objects set with all burners at 0 heat. The code segment below creates two `Stove` objects, `s1` and `s2`. After that, write java code that does the following:

    - Crank up the heat on burner 2 of `s1` (heat setting of 10).
    - Print out the setting for burner 4 on `s2`.
    - Turn off burner 2 on `s1`.

```
Stove s1, s2;

s1 = new Stove();
s2 = new Stove();
```

(back)scratch page

READ THIS STATEMENT AND SIGN BELOW:

I have read over this exam, if time permitted me, and have completed it to the best of my independent ability. I have adhered to the instructions listed on the cover sheet of this exam. I have looked at and am turning in all 12 pages of the exam (counting this page).

SIGNED: