University of Cape Town

Department of Computer Science

COURSE : CSC305H

MODULE : COMPILERS

TEST : 2

DATE : 17 AUGUST 2005

TOTAL MARKS : 35

INSTRUCTIONS TO STUDENTS

- Answer ALL questions

Question One – Symbol Tables and Activation Records [10]

1. Discuss, with diagrams, how a hash table in a functional symbol table is modified when a scope is opened and then when a new variable definition in the new scope overrides an existing one from a previous scope. [5]

2. Consider the following Pascal-like program with static scoping:

```
program test;
var a : integer;

   procedure proc1;
   var b : integer;
   begin (* POINT1 *)
   end;

   procedure proc2;
   var a, c : integer;
   begin
      (*POINT2 *) proc1; (*POINT3 *)
   end;

begin
   (*POINT4 *) proc2; (*POINT5 *)
end.
```

Which variables (state variable name and subprogram name) are in scope at each of the 5 points? [5]

Question Two – Intermediate Representation [10]

1. Discuss 2 advantages of intermediate representations. [3]

2. Assuming the IR tree language in the attached pages, convert the following program fragment to an equivalent IR tree. (Assume a/b are stack frame variables at offset k0/k1 from the frame pointer special temporary fp) Provide the final tree and do not use the Nx/Cx/Ex expression types/objects. [4]

```
if (a<b) a=1 else b=1;
```

3. Convert the following tree into its canonical form by applying transformations from the attached list. Show the result after each transformation. [3]

JUMP (ESEQ (LABEL L1, MEM (ESEQ (LABEL L2, TEMP t))))

Question Three – Instruction Selection [8]

1. What is the difference between an optimal and optimum tiling? Give one example of an algorithm in each class, and state what the Big-O complexity of each algorithm is. [4]

2. Using the attached instruction set, apply the Maximal Munch tiling algorithm to the following IR tree. Show the tiled tree and list the instructions generated. [4]

MEM (PLUS (CONST a, TIMES (CONST b, MEM (PLUS (CONST c, CONST d)))))

Question Four – Register Allocation [7]

1. Use the iterative liveness analysis algorithm to calculate the live-in and live-out sets for each of the following statements in a program. Show succ, use, def, out and in sets. [7]

```
a = c
b = d
if ( a < b )
   then m = b;
   else m = a;
return m;
```

Hint: The relevant formulae are:

$$out[n] = \bigcup_{s \in succ[n]} in[s]$$

$$in[n] = use[n] \cup (out[n] - def[n])$$