University of Cape Town

Department of Computer Science

CSC3003S Class Test

2007

Marks : 20

Time : 45 minutes

Instructions:

- Answer all questions from Section A and 3 questions from Section B.
- Show all calculations where applicable.

Section A [Answer Question ONE – this is compulsory]

Question 1

a) What is the purpose of the context-sensitive analysis phase of semantic anlysis? [1]

check for errors due to surrounding context e.g., type checking

b) Besides context-sensitive analysis, what is the other task carried out by the semantic routines of a compiler? [1]

generate machine code

c) Discuss 1 advantage and 1 disadvantage in using intermediate representations in compilers. [2]

adv: separation of front/back ends, easier to apply optimisations to, etc.

disad: additional processing, mismatch between IR and real MC, etc.

d) During the process of generating machine code, when should registers be used instead of memory? [1]

as much as possible

Section B [Answer 3 questions ONLY]

Question 1: Symbols and Types

a) What is scope and why is it desirable to have multiple scopes? [2]

where names can be referenced [1]

makes programmers life easier – less context to deal with [1]

b) What is the difference between name and structural type equivalence? Discuss with an example. [3]

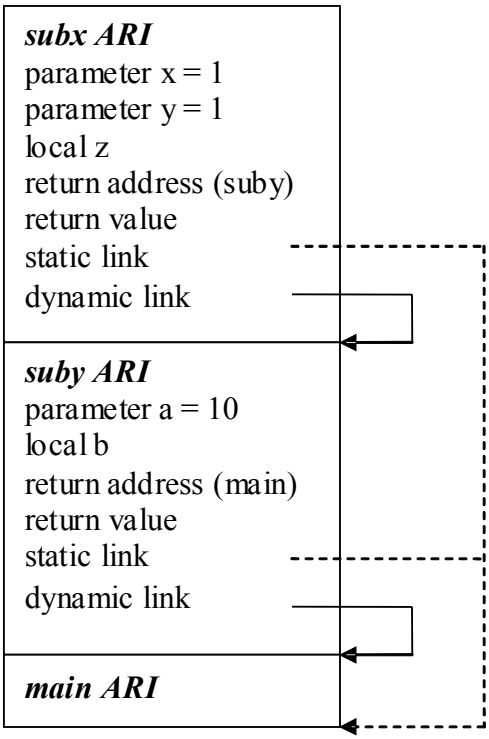
name equivalence checks names of types only while structural equivalence checks structure of types irrespective of their names. [1] example: typedef int anint; int x, z; anint y; x and z are name equivalent while x and y are structurally equivalent [2]

Question 2: Activation Records

a) What is an activation record? [1]

the layout of data to support subprogram invocation

b) Write a skeletal program that will result in the following activation record. [4]




```
main ()
{
  int subx ( int x, int y )
  {
    int z;
    return ...
  }
  int suby ( int a )
  {
    int b;
    subx (1, 1)
    return ...
  }
  suby (10);
}
```

Minus one mark for each major error
or
Nesting=1, Parameters=1, Locals=1, Subprogram call statements=1

Question 3: Code Generation

a) Using the attached IR language, convert the following C-like expression to an unoptimised IR tree. Assume **a** and **b** are stack variables at offsets *k_a* and *k_b* respectively from the frame

pointer TEMP(FP). Provide the final tree and do not use the Nx/Cx/Ex expression types/objects. [3]

```
a = ( b + 5 ) * 10
```

MOVE(MEM(+(TEMP(FP),CONST(k_a))),
**(CONST(10),+(MEM(+(TEMP(FP),CONST(k_b))),CONST(5))))*

Minus one mark for each major error.

b) State the formula used to calculate memory offsets for element A[i, j] of a 2-dimensional array. Assume either row-major or column-major order. [2]

A : array [L1..U1, L2..U2] of type T

*Offset = [(U2-L2+1) * (i-L1) + (j-L2)] * sizeof (T)*

Question 4: Basic Blocks and Traces

a) What is a basic block? [2]

a linear sequence of statements starting with a label and ending with a jump

b) What benefit is there in rearranging basic blocks into traces? [1]

we could possibly eliminate redundant jumps

c) In preparation for instruction selection, what modifications do we need to make to code where a CJUMP is followed by a label other than its true and false labels? Illustrate with an example. [2]

Add in an immediately-following jump to the false label

CJUMP (cond, a, b, lt, lf); LABEL lfprime; JUMP (NAME lf)