Two hours

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Compilers

Date: Wednesday 21st May 2014

Time: 14:00 - 16:00

Please answer any THREE Questions from the FIVE Questions provided

This is a CLOSED book examination

The use of electronic calculators is NOT permitted

[PTO]

1. a) Explain briefly what a symbol table is. Your answer should mention when in compilation symbol table entries are normally created.

(3 marks)

b) Consider the following C program:

```
1. int main()
2. {
3.   char *p;
4.   register int x;
5.
6.   scanf("%d",@x); scanf("%d",&y);
7.   if (x > y) then { x=y; }
8.   y=x+1_007;
9.   y+=(double)(p);
10.  printf("%d\N", y);
11. }
```

When compiling this program a certain compiler produces the following error messages:

```
Line 6: syntax error at '@' token
Line 6: ‘y’ undeclared (first use in this function)
Line 7: ‘then’ undeclared (first use in this function)
Line 7: parse error before '{' token
Line 8: underscore in number
Line 9: pointer value used where a floating point value
was expected
Line 10: warning: unknown escape sequence '\N'
```

Indicate which compilation phase is most likely to report each of the above error messages. Justify your answer.

(7 marks)

c) Briefly describe the output of each major compilation phase for the assignment statement `x=y*z+3+2`, where `x`, `y`, `z` are real numbers. State any assumptions you make.

(6 marks)

d) Explain why the `register` keyword in C, which tells the compiler to store a variable in a CPU register (for example, see line 4 in b) above), might have been a good idea when the language was first developed, but is probably not a good idea nowadays.

(4 marks)

2. a) Given a Non-Deterministic Finite Automaton (NFA) with several final states explain how to convert this to an NFA with only one final state.

(3 marks)

b) Explain whether it is possible to write regular expressions to recognise:

(i) any even integer

(ii) any binary number which contains as many 0s as 1s

(iii) any real number which contains the same number of digits in both the integer part and the decimal part (e.g., 12.45 is such a number, but 3.14 is not)

(iv) any real number with an exponential part (e.g., 3.45E2)

(v) any string of lowercase letters in which the letters are in ascending lexicographic order (for example, amrz or delos are such strings but anno or agora are not) Justify your answers.

(5 marks)

c) Draw a minimum state DFA for the regular expression (a | b)* a (a | b)

(7 marks)

d) An IP address consists of four integers each with a value between 0 and 255 (including 0 and 255), which are separated by a dot. For example, 34.0.23.54 is a valid IP address whereas 52.256.1.7 is not a valid IP address. Write a regular expression to recognise a valid IP address.

(5 marks)

[PTO]

3. Consider the following context-free grammar for variable declarations in a Pascal like language:

1. Goal → Var_Decl
2. Var_Decl → var Decl_List
3. Decl_List → Decl ; Decl_List
4. Decl_List → Decl ;
5. Decl → Id_List : Id_Type
6. Id_List → Id_List , identifier_name
7. Id_List → identifier_name
8. Id_Type → integer
9. Id_Type → real
10. Id_Type → boolean
11. Id_Type → char

a) Derive a leftmost derivation for the string

```
var x,y:integer; z:boolean;
```

and show the corresponding parse tree..

(6 marks)

b) Transform this grammar so that it can be used to construct a top-down predictive parser with one symbol of lookahead.

(6 marks)

c) Explain how a bottom-up shift-reduce parser works and what a shift/reduce conflict is. Show, in full detail, the steps that a shift-reduce parser would follow to parse the string:

```
var x,y:integer;
```

For each step your answer should show the contents of the stack, what the next input is and the action that is taken.
[NB: you should consider the original grammar, not the grammar derived in b) above]

(8 marks)

4. a) Convert the following, C-like code fragment into stack machine code. State your assumptions.

```
      if (x > 0) { x = y + 1; }
```

(3 marks)

b) The following shows two different approaches to generate code for the assignment statement: `x=a*b+c*d+e*f`
Explain the difference between the two approaches highlighting the advantages of each approach.

```
      load r1, @a                  load r1, @a
      load r2, @b                  load r2, @b
      mult r1,r1,r2                load r3, @c
      load r2, @c                  load r4, @d
      load r3, @d                  load r5, @e
      mult r2,r2,r3                load r6, @f
      add  r1,r1,r2                mult r1,r1,r2
      load r2, @e                  mult r3,r3,r4
      load r3, @f                  mult r5,r5,r6
      mult r2,r2,r3                add  r1,r1,r3
      add  r1,r1,r2                add  r1,r1,r5
```

(3 marks)

c) Sketch an algorithm that you would incorporate in a compiler to generate code that implements the multiplication of an integer variable with an integer constant as a sequence of shifts and additions.

(5 marks)

d) Discuss two compiler optimisations, whose impact on program performance is expected to be higher when compiling Java than when compiling C.

(4 marks)

e) A certain C compiler performs the following optimisation:

```
/* before optimisation */     /*after optimisation */
for (i=1;i<1000;i++)            for (i=1;i<1000;i++)
  for(j=1;j<1000;j++)             for(j=1;j<1000;j++)
     a[j][i]+=1;                     a[i][j]+=1;
```

Why is this an optimisation? What would stop the compiler from applying this optimisation if the assignment statement of the original code was as follows?

```
      a[j][i]+=a[j-1][i+1];
```

(5 marks)

[PTO]

5. a) Explain the difference between register allocation and register assignment.

(2 marks)

b) In the context of register allocation, explain what spilling is. Suggest two approaches that can be used to decide what to spill.

(4 marks)

c) The following piece of code makes use of 7 temporary variables.

```
1. a = 1
2. b = a + 3
3. c = a + b
4. d = b + 5
5. e = c + 7
6. f = e + d
7. return f
```

Draw an interference graph and apply a graph colouring algorithm of your choice to find the fewest number of variables that may be used to write this piece of code. Show the resulting code. State any assumptions you make.

(6 marks)

d) Apply list scheduling to generate a schedule for the following basic block on a processor that can issue up to two instructions per cycle. Your answer should show the precedence graph, explain how you set priorities and show the schedule. Assume that all instructions have a latency of 1 cycle except `mult` that has a latency of 2 cycles.

```
1. mult r1, r1, 7
2. mov  r2, 1
3. mult r3, r3, 12
4. add  r4, r1, r2
5. add  r5, r2, 4
6. add  r6, r2, r3
7. add  r7, r5, 6
8. add  r8, r5, 9
```

(5 marks)

e) A critical aspect of list scheduling is the mechanism for setting priorities and breaking ties when several operations with the same priority are ready at the same cycle. Using as an example the schedule generated in d) above, demonstrate how different mechanisms for setting priorities and breaking ties can lead to different schedules.

(3 marks)

END OF EXAMINATION