# 國立交通大學試題紙

九十八學年度第二次
博士班資格考

科目：編譯器設計(A)　　日期： 99 年 7 月 29 日　第 1 頁 共 2 頁

請"✓"明　✓不可看書　可看書

* 請將答案依題號順序寫入答案卷

答題時字跡需工整，否則不予計分。Write your answers legibly; otherwise you will get zero score.

**Basic questions: Please answer all three questions in this part.**

1. (20 points) Find the LL(1) parse table for the following grammar:

$S \rightarrow V$

$S \rightarrow W$

$V \rightarrow a\ X\ b$

$W \rightarrow c\ X\ d$

$X \rightarrow$

You need to show the detailed process of constructing the parse table.

2. (20 points) What are the possible conflicts when we use regular expressions to specify the tokens in a programming language? For each of the these conflicts, write down plausible (simple or complex) solutions.

3) Code Generation

a) In a compiler, what data structures are commonly used to represent an expression?

b) An expression may contain some Common Sub-Expressions (CSE), for example, in the following expression

(a+b) * (a+b),

a+b is a CSE.

The result of a CSE can be kept in a register so that subsequent CSEs can directly use the saved result and avoid redundant computation.

How can a compiler capture such CSEs.

c) CSE may cross expressions, for example, in the following statements

S1: x = (a+b)*c;

S2: y = (a+b)*d;

(a+b) is a CSE across statements. How is catching such CSEs different from methods used in (b)?

# 國立交通大學試題紙

九十八學年度第二次
博士班資格考

科目：編譯器設計(B)　　日期：99年7月29日　第1頁 共2頁

請"✓"明　✓不可看書　可看書

* 請將答案依題號順序寫入答案卷

答題時字跡需工整，否則不予計分。Write your answers legibly; otherwise you will get zero score.

**Advanced questions: Please answer two out of the following four questions in this part.**

4. (20 points) Please design a data structure to represent types in common programming languages, such as C or Java. Assume that there are basic types (integer and real) and structured types (pointers, arrays and records). The array could have as many dimensions as the programmers wishes.

5. (20 points) Given a sequence of assembly-like instructions and $n$ registers, how can we determine if the variables used in the sequence can be mapped to the $n$ registers without register spilling? Obviously, we can try all possible mappings exhaustively. Can you find a cleverer algorithm?

```
 1.  doubleSum(x, y) {
 2.    initVal = x  // 1st parameter
 3.    limit = y    // 2nd parameter
 4.    sum = initVal
 5.    i = 1
 6.    temp1 = i <= limit
 7.    while (temp1) {
 8.      temp2 = sum + 1
 9.      sum = temp2
10.      temp3 = i + 1
11.      i = temp3
12.      temp1 = i <= limit    }
13.   temp4 = 2 * sum
14.   return temp4  // return value
15.   }
```

Figure A sample program.

Advanced questions (Optional parts)

6. Compiling Loop Structures
   a) The **For** loops are more commonly used than the **WHILE** loops. Compiling a **For** loop is more difficult than compiling a while loop. Explain what are the difficulties involved.
   b) A **For** loop in C uses the following grammar rule:
      **STMT: FOR "(" assignment_expression_list ";" relop_expression_list ";" assignment_expression_list ")" STMT**
      Please write down action routines associated with the above production rule and a pseudo code sequence to illustrate what code sequence to generate from the action routines of such a **For** loop structure.
7. JIT translation
   Java virtual machines often include a JIT (Just-in-Time) Translator.
   a) How is a JIT different from a typical compiler?
   b) For modern micro-processors, compiler optimizations are increasingly more important. How does a JIT compiler perform optimizations? What are the pros and cons of JIT optimizations?