University of Waterloo

University of Waterloo Midterm Examination

Fall 2008

Student Name:	_____
UW Student ID Number:	_____ UW Userid: _____

Course Abbreviation and Number:	*CS 241*
Course Title:	*Foundations of Sequential Programs*
Section(s):	*001, 002, and 003*
Instructor(s):	*Ondřej Lhoták and J.P. Pretti*

Date of Exam:	*October 30, 2008*
Time Period:	Start time: *4:30 PM* End Time: *6:30 PM*
Duration of Exam:	*2 hours*
Number of Exam Pages:	*13 (including this cover sheet)*
Exam Type:	*Closed Book*
Additional Materials Allowed:	*MIPS and MERL Reference Sheets* *Number/Character Conversion Reference Sheet*

If you believe there is an error in a question, please bring it to the attention to a proctor. He or she will report your concern to an instructor. Otherwise, clearly state any extra assumptions you feel necessary to answer a question.

Marking Scheme

Question	Max	Score
1	4	
2	4	
3	13	
4	4	
5	11	
6	8	
7	6	
8	6	
9	5	
10	4	
TOTAL	65	

1. [4 marks] **Short Answers**

   Answer each of the following using no more than one or two sentences.

   (a) What is the minimum two's complement number that can be represented in four bits?

   (b) What is the maximum two's complement number that can be represented in four bits?

   (c) What does 10100100 represent?

   (d) How does an understanding of formal languages help in the creation of a compiler?

2. [4 marks] **MIPS Machine Language Programming**

   Translate the following MIPS assembly language program to MIPS machine language. Express your answer in hexadecimal in the spaces provided.

```
.word -2          0x ___ ___ ___ ___ ___ ___ ___ ___

sub $9, $1, $7    0x ___ ___ ___ ___ ___ ___ ___ ___

sw $3, 8($9)      0x ___ ___ ___ ___ ___ ___ ___ ___

jr $31            0x ___ ___ ___ ___ ___ ___ ___ ___
```

3. [13 marks] **MIPS Assembly Language Programming**

Write MIPS assembly language programs to accomplish the following tasks. You may freely alter the values of any register without restoring them to their original contents.

(a) [7 marks] Register 1 holds the address of the beginning of an array of 32-bit unsigned integers. Register 2 holds the number of elements in this array. Find the minimum element of the array and store it in register 3. Then return. Assume the array is not empty.

(b) [2 marks] Register 1 holds a valid address of a word in memory (that is not the last word in memory). Swap the contents of memory at this address with the contents of the following word in memory. Then return.

(c) [4 marks] Register 1 holds the address of the beginning of an array of 32-bit integers. The array encodes a tree exactly as described in Assignment 2:

Each node of the tree is encoded in three consecutive elements (words) of the array: a two's complement integer stored at the node, the node's left child, and the node's right child. Each child is specified as the array index of the first element of the child node. The integer -1 indicates that a node does not have a left or right child. For example, the following tree:

```
    77
   /  \
22     -8
      /  \
   -36    999
```

could be encoded by the array to the right.

```
A[0]  = 77
A[1]  = 3
A[2]  = 6
A[3]  = 22
A[4]  = -1
A[5]  = -1
A[6]  = -8
A[7]  = 9
A[8]  = 12
A[9]  = -36
A[10] = -1
A[11] = -1
A[12] = 999
A[13] = -1
A[14] = -1
```

Compute the height of the right subtree of the root node and place it into register 3. Then return. (For the example tree described above, you would compute the height of the subtree whose root node has value 8. The height of this right subtree is 2.) Assume the right subtree is not empty.

Call (but do not write!) a procedure with the label height which you may assume has already been assembled and will be linked with the main program you write. The height procedure expects to find the memory address of the beginning of the array containing the tree in register 1 and the index of the node comprising the root of a subtree in register 2. The procedure returns its result, the height of the specified subtree, in register 3.

4. [4 marks] **MIPS Hardware Architecture**

The values of some of the registers in a MIPS CPU and part of the associated memory are shown below in hexadecimal.

```
$1 = 0x11111111  $2 = 0xffffffff  $3 = 0x33333333  pc = 0x00000000
```

The contents of the first two words of memory are:

Address	Contents
0x00000000	0x00001814
0x00000004	0x03e00008
...	...

Starting in this configuration, the CPU performs one iteration of the fetch-increment-execute cycle (also known as the machine cycle). For each of the following registers, write its value after one iteration of the cycle is completed.

```
$1 = 0x __ __ __ __ __ __ __ __        $2 = 0x __ __ __ __ __ __ __ __

$3 = 0x __ __ __ __ __ __ __ __        pc = 0x __ __ __ __ __ __ __ __
```

5. [11 marks] **Assemblers**

(a) [5 marks]

Consider the following MIPS assembly language program, to be assembled starting at memory address 0.

```
lis $4
a: b:
c:

.word d
d: add $1, $2, $3
; this is a comment
e:
```

Complete the symbol table below (produced and used by an assembler) for the above code. Express the values of symbols in hexadecimal.

symbol	*value*
a	0x
b	0x
c	0x
d	0x
e	0x

(b) [3 marks] List all sequences of one or more strings from the set{ “42”, “label:”, “.word”,“;hi” } that are valid lines of MIPS assembly code when separated by spaces. Each valid line, by itself, should be an assembly language program that would be accepted by the MIPS assembler. No string may appear more than once in a given line.

(c) [3 marks] Consider the following MIPS assembly language program.

```
a: beq  $5,  $6,  c
b: lis  $4
     .word  d
c: bne  $7,  $8,  a
d: .word  0xf
```

Write an equivalent MIPS assembly language program that does not contain any labels. Both assembly language programs, when assembled, must result in identical machine language programs. You are not required to show the machine language version of the program.

6. [8 marks] **Linkers and Loaders**

(a) [4 marks] The following is a MERL file (shown in hexadecimal) encoding a short MIPS machine language program.

```
10 00 00 02
00 00 00 24
00 00 00 1c
00 00 38 14
00 00 00 18
00 e0 00 09
03 e0 00 08
00 00 00 01
00 00 00 10
```

The above MERL file is loaded starting at memory address 0x88, then relocated so that it can execute correctly at this address. The resulting contents of each memory location, in hexadecimal, are shown below. Fill in each blank with the correct hexadecimal digit.

Address	Contents
0x00000088	0x 1 0 0 0 0 0 0 2
0x0000008c	0x 0 0 0 0 ___ ___ ___ ___
0x00000090	0x 0 0 0 0 ___ ___ ___ ___
0x00000094	0x 0 0 0 0 ___ ___ ___ ___
0x00000098	0x 0 0 0 0 ___ ___ ___ ___
0x0000009c	0x 0 0 e 0 ___ ___ ___ ___
0x000000a0	0x 0 3 e 0 ___ ___ ___ ___
0x000000a4	0x 0 0 0 0 ___ ___ ___ ___
0x000000a8	0x 0 0 0 0 ___ ___ ___ ___

(b) [4 marks] Consider the following MIPS assembly language program:

```
   .export a
   .word a
b: .word 0xabc
a: bne $1, $2, b
```

The following MERL file, in hexadecimal, encodes the program so that it may be linked with other programs and loaded at any address. Fill in each blank below with the correct hexadecimal digit.

Address	Word
0x00000000	0x 1 0 0 0 0 0 0 0 2
0x00000004	0x ___ ___ ___ ___ ___ ___ ___ ___
0x00000008	0x ___ ___ ___ ___ ___ ___ ___ ___
0x0000000c	0x 0 0 0 0 0 0 1 4
0x00000010	0x 0 0 0 0 0 a b c
0x00000014	0x 1 4 2 2 f f f e
0x00000018	0x ___ ___ ___ ___ ___ ___ ___ ___
0x0000001c	0x ___ ___ ___ ___ ___ ___ ___ ___
0x00000020	0x ___ ___ ___ ___ ___ ___ ___ ___
0x00000024	0x ___ ___ ___ ___ ___ ___ ___ ___
0x00000028	0x ___ ___ ___ ___ ___ ___ ___ ___
0x0000002c	0x ___ ___ ___ ___ ___ ___ ___ ___

7. [6 marks] **DFAs**

For the following, assume the alphabet is $\Sigma = \{a, b, c\}$.
You are NOT required to draw the implicit error state in any DFA bubble diagrams.

(a) [2 marks] Draw a DFA (bubble diagram) that recognizes all words in which no b appears earlier than any a. For example, ϵ, aa, $cacbcc$ and $cbbb$ should be accepted but ba and $caccbba$ should be rejected.

(b) [4 marks] Draw a DFA (bubble diagram) that recognizes all words in which no b appears earlier than any a and there are an even number of $c's$.

8. [6 marks] **Formal Languages**

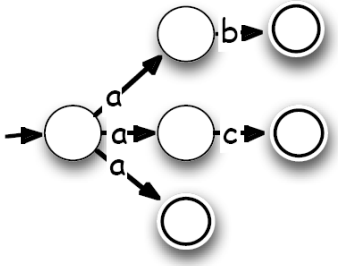
(a) [2 marks] Let M be a DFA where $\Sigma = \{x, y\}$, $Q = \{S, T, E\}$, $q_o = S$, $A = \{S, T\}$, $\delta(S, x) = T$, $\delta(S, y) = T$ and $\delta(q, \sigma) = E$ for all other $q \in Q$ and $\sigma \in \Sigma$. What is $L(M)$, the language accepted by M?

(b) [4 marks] Give pseudocode for a recognizer for the language defined by a DFA with states S, initial state i, final states F and transition function T. Assume the input string is $z = z_1 z_2 \ldots z_n$.

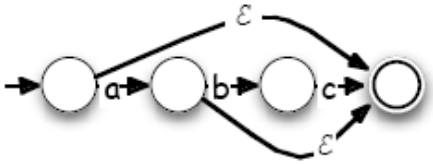
9. [5 marks] **NFAs and Regular Expressions**

   For the following, assume the alphabet is $\Sigma = \{a, b, c\}$.

   (a) List the words in the language accepted by the following NFA.



   (b) List the words in the language accepted by the following ϵ-NFA.

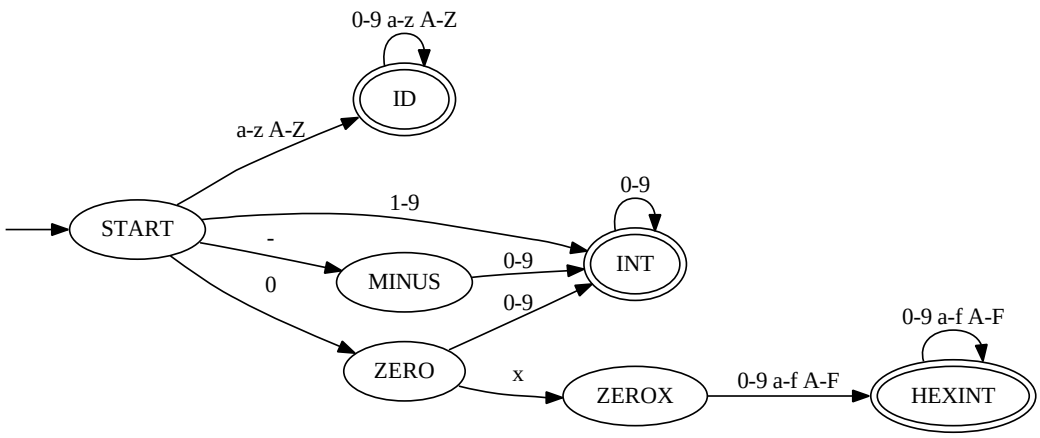


   (c) List the words in the language specified by the regular expression $(a|b)(c|\epsilon)$.

   (d) Give a regular expression recognizing all words with an odd number of a's.

10. [4 marks] **Scanning**

The following DFA specifies a language of valid tokens.



The *simplified maximal munch algorithm* is executed using the above DFA on the following input string: `0xa0xb0x12c-30xcd`

List the lexemes of the tokens that the algorithm outputs.

Note: the `0` symbols in the input string are all zeros, *not* the letter O.