Points on this page: 10

Q1a-d 10

1. (21 points) Short answers:

(a) (1 point) True or False? If an unsigned integer variable `x` contains the encoding of a MIPS instruction, then before an assembler can output the bytes of `x`, it must first convert `x` to hexadecimal.

(b) (1 point) True or False? If `x` and `y` are unsigned integer variables, then the expressions `x+y` and `x | y` always have the same value.

(c) (1 point) True or False? If `x` and `y` are unsigned integer variables, then the expressions `x+y` and `x | y` never have the same value.

(d) (7 points) Give the symbol table that an assembler would construct for the following assembly language program (give addresses in hexadecimal):

```
  barney:
0 sw $4, -4($30)
4 lis $4

8 four: eight: .word 4
C sub $30, $30, $4

  dino:

  pop: ; must come off
10 add $30, $30, $4
14 lw $4, -4($30)
18 wilma: jr $31
1C betty:
```

Label	Address in hexadecimal
barney	
four	
eight	
dino	
pop	
wilma	
betty	

Q1e-g 7

(e) (3 points) Give the 6-bit two's complement binary representation of the following values:

- -1 =
- 22 =
- -32 =

(f) (4 points) What MIPS instruction (other than a .word assembler directive) is represented by the encoding `0x03db382b`?

(g) (4 points) A numeric constant (say, 241) is located somewhere in a MIPS file, and its location is labelled:

```
courseNumber:    .word 241
```

Give a fragment of MIPS code to load the value of this constant into register `$3`. For the example above, once your given code has executed, `$3` should contain the number `241`. Your code should work no matter what the value of the constant is (i.e., you must not not hard-code the number 241), but you may assume that the label is called `courseNumber`.

Points on this page: 0

Q2text

2. (16 points) Assembly Language Programming

(a) (10 points) Create a MIPS assembly language program to determine if an array of unsigned integers, A, is sorted in ascending order (allowing for repeated values). That is for each pair of neighbours, A[i] $\leq$ A[i+1]. Both 1, 2, 5 and 1, 2, 2, 3 have this property.

Specification: The input is in $1 (which holds the address of the beginning of the array) and $2 (which holds the number of elements in the array). If A is sorted in ascending order then store a 1 in register $3, otherwise store a 0 in $3. The input is guaranteed to be valid and the size of the array is at least two. The input can be modified by your program.

Marking: Your solution will be marked for correctness, efficiency and documentation.

Use of Registers: You **must use** registers 4–9 for the purposes stated below and if you use registers $10–$12 clearly indicate what you use them for. You may assume that $4 and $5 already have the stated values. You may not need all the registers for your solution.

```
;; $1 address of the array, A          $2 size of the array
;; $3 answer                           $4 the constant 1
;; $5 the constant 4                   $6 i
;; $7 the value stored at A[i]         $8 the value stored at A[i+1]
;; $9 result of comparing two array values
```

a) Document any additional registers that may be needed.

```
;; $10                                 $11

;; $12                                 $13
```

b) Write the program (you may continue to the next page).

Q2a 7

(Page intentionally blank for continuation of previous question)

Points on this page: 6

Q2b 6

(b) (6 points) Suppose you are given two procedures, `f` and `g`, written in MIPS assembly. Both of these two procedures take an input parameter in register $1, and return their output in register $3. Write a complete MIPS assembly main program that places the value of `f(g(15))` into register $4. You may assume that the definitions of `f` and `g` will be supplied (with labels `f:` and `g:`, respectively) immediately after your main program.

Q3 8

3. (9 points) Regular Languages

(a) (3 points) Give a regular expression for the language: Strings over $\{a, b\}$ that start and end with the same character.

(b) (4 points) Give a DFA for the following language: Strings over $\{a, b\}$ that contain at least one a, and no more than two b's.

(c) (2 points) Give a DFA for the following language: $\{\varepsilon\}$.

Points on this page: 10

Q4 10

4. (10 points) Context Free Grammars

Consider the following context-free grammar, with start symbol S:

$$\begin{aligned} S &\to a\,T\,T \\ S &\to a \\ T &\to c\,T \\ T &\to S \end{aligned}$$

(a) (3 points) Show, using a leftmost derivation, that this grammar generates *acacca*.

(b) (2 points) Explain why this grammar *cannot* generate *aaaa*.

(c) (5 points) Show that this grammar is ambiguous, by finding a string with at least two parse trees. Give both parse trees for this string.

Q5a-c 5

5. (16 points) LL(1) Parsing

Consider the following grammar:

1) S'→ ⊢ S ⊣
2) S → A
3) S → B
4) A → a B
5) A → C x
6) A → ε
7) B → b
8) B → D f
9) C → e C
10) C → ε
11) D → d
12) D → ε

(a) (1 point) Is the non-terminal S nullable?

(b) (2 points) What is First(B), i.e., the First set for the non-terminal B?

(c) (2 points) What is Follow(B), i.e., the Follow set for the non-terminal B?

Points on this page: 4

Q5d 2

(d) (4 points) Complete the LL(1) predict table for this grammar by filling in the Predict sets for the non-terminals S, A and B. The rest of the rows have already been filled. Note: this grammar is indeed LL(1). The grammar is restated below for convenience.

1) S' → ⊢ S ⊣
2) S → A
3) S → B
4) A → a B
5) A → C x
6) A → ε
7) B → b
8) B → D f
9) C → e C
10) C → ε
11) D → d
12) D → ε

see solution -2

Non-terminal	⊢	⊣	a	x	b	f	e	d
S'	{ 1 }	{ }	{ }	{ }	{ }	{ }	{ }	{ }
S								
A								
B								
C	{ }	{ }	{ }	{ 10 }	{ }	{ }	{ 9 }	{ }
D	{ }	{ }	{ }	{ }	{ }	{ 12 }	{ }	{ 11 }

Q5ef 5

(e) (5 points) Trace through a top-down parse of the string ⊢ a b f ⊣, by filling in the table below. Each action should be one of a rule being applied, a symbol being matched or the parser accepting or rejecting the input. (There is plenty of room in the table and you might have some rows left blank)

Read Input	Unread Input	Stack	Action

(f) (2 points) If we change rule 8 to `B -> D b`, then the grammar is no longer LL(1). Explain why.

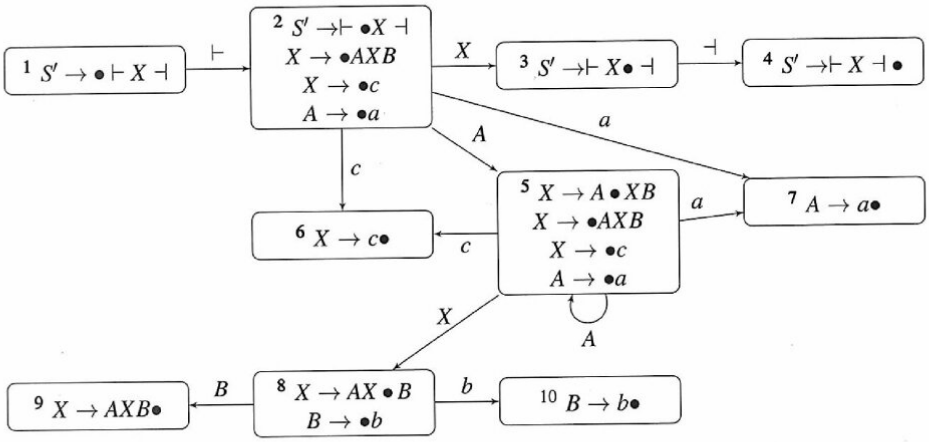
no answer -2

Points on this page: 0

Q6text

6. (10 points) Bottom-up Parsing

Using the following LR(0) machine, give the step-by-step parse for input: ⊢ acb ⊣. For each Action, specify shift or reduce. For a shift, specify the symbol being shifted. For a reduce, specify the production being reduced.



Use the table on the next page to write your answer starting at the Action column in row 1. (There is plenty of room in the table and you might have some rows left blank)

Q6 10

Unread Input	Symbol Stack	State Stack	Action
⊢acb⊣	ε	1	Shift ⊢
acb⊣	⊢	12	Shift a
cb⊣	⊢a	127	Reduce A → a•
cb⊣	⊢A	125	Shift c
b⊣	⊢Ac	1256	Reduce X→c•
b⊣	⊢AX	1258	shift b
⊣	⊢AXb	12581 0	Reduce B → b•
⊣	⊢AXB	12589	Reduce X → AXB•
⊣	⊢X	123	Shift ⊣
ε	⊢X⊣	1234	Reduce S' → ⊢X⊣•
ε	S'	1	Accept

Points on this page: 0

extra

This page is intentionally left blank for your use. Do NOT remove it from your booklet. If you require more space to answer a question, you may use this page, but you **must clearly indicate** in the provided answer space that you have done so.