CSCI 2500 — Computer Organization Homework 5 (document version 1.0) Pipelining in MIPS

Overview

- This homework is due by 11:59:59 PM on Tuesday, December 4, 2018.
- This homework is to be completed **individually**. Do not share your code with anyone else.
- You **must** use C for this homework assignment, and your code **must** successfully execute on Submitty to obtain full credit.

Homework Specifications

For this individual homework assignment, you will use C to implement a simulation of MIPS pipelining. A verget graph better, there is contagned to the pipeline, i.e., i

For your simulation, you are required to support the add, sub, and, or, 1w, and sw instructions; note that some of the attractions which is fine. Worg necifically, you must simulate (and output) how a given sequence of instructions would be pipelined in a five-stage MIPS implementation.

Do not implement for Ardin W singlettrat powcoder

You can assume that each given instruction will be syntactically correct. You can also assume that there is a single space character between the instruction and its parameters. Further, each parameter is delimited by a comma or parentheses. Below are a few example instructions that you must support:

```
add $t0,$s2,$s3
add $t1,$t3,73
or $s0,$s0,$t3
lw $a0,12($sp)
sw $t6,32($a1)
```

Required Command-Line Argument

Your program must accept one command-line argument as input. This argument (i.e., argv[1]) specifies the input file containing MIPS code to simulate. You may assume that no more than five instructions are given in the input file. And note that each instruction will end with a newline character (i.e., '\n').

Required Output

For your output, you must show *each cycle* of program execution. Each cycle will correspond to a column of output. Initially, each column is empty, indicated by a period (i.e., '.'). Use TAB characters (i.e., '\t') to delimit each column. And assume that you will have no more than nine cycles to simulate.

Recall that a *data hazard* describes a situation in which the next instruction cannot be executed in the next cycle until a previous instruction is complete. Your code should be able to detect when it is necessary to insert one or more "bubbles" (see Section 4.7 of the textbook and corresponding lecture notes for more details).

More specifically, you will need to properly handle data hazards by adding nop instructions as necessary. Show these cases by indicating an asterisk (i.e., '*') in the appropriate columns and adding the required number of nop instructions. To ensure proper formatting, add an extra TAB character after the nop.

On the next few pages, we present a few example runs of your program that you should use to better understand how your program should work, how you can test your code, and what output formatting the Significant Project Exam Help

https://powcoder.com

Add WeChat powcoder

The first example (i.e., ex01.s) includes no data hazards.

START OF SIMULATION

CPU	Cycles ===>	1	2	3	4	5	6	7	8	9
add	\$s1,\$s0,\$s0	IF								
add	\$t2,\$s0,\$s5	•	•	•			•	•	•	
add	\$t4,\$s3,70		•	•	•	•		•		
	Cycles ===>		2	3	4	5	6	7	8	9
	\$s1,\$s0,\$s0		ID	•	•	•	•	•	•	•
	\$t2,\$s0,\$s5	•	IF	•	•	•	•	•	•	•
add	\$t4,\$s3,70	•	•	•	•	•	•	•	•	٠
CPU	Cycles ===>	1	2	3	4	5	6	7	8	9
	\$s1,\$s0,\$s0		ID	EX						
	\$t2,\$s0,\$s5		IF	ID						
	\$t4,\$s3,70			IF						
	Δς	sian	men	t Dro	niect	t ₅ Exa	am I	Jeln		
CPU	Cycles ===>	argii		3 1 1		5		j CIP	8	9
add	\$s1,\$s0,\$s0	IF	ID	EX	MEM					
add	\$t2,\$s0,\$s5	. 1	IF /	/ID	EX 1					
add	\$t4,\$s3,70	· ntt	ps://	POW	/@Od	er.co	om	•		
			1	1						
	Cycles ===>		2	3	4	5	6	7	8	9
	\$s1,\$s0,\$s0	/ 🔪 /	XI K	/EX (~1)	MEM	WB	ode	r.	•	•
	\$t2,\$s0,\$s5	. A	l <u>(</u> d VV	10 CI	lat p	MEMV	Ouc	L.	•	•
add	\$t4,\$s3,70	•	•	IF	ID	EX	•	•	•	•
CDII	Cycles ===>	1	2	3	4	5	6	7	8	9
	\$s1,\$s0,\$s0		ID	EX	4 MEM	WB	0	1	0	9
			IF	ID	EX	MEM	WB	•	•	•
	\$t2,\$s0,\$s5	•	IF		ID	EX		•	•	•
auu	\$t4,\$s3,70	•	•	IF	עד	ĽΛ	MEM	•	•	•
CPU	Cycles ===>	1	2	3	4	5	6	7	8	9
add	\$s1,\$s0,\$s0	IF	ID	EX	MEM	WB				
add	\$t2,\$s0,\$s5		IF	ID	EX	MEM	WB			
add	\$t4,\$s3,70			IF	ID	EX	MEM	WB		

END OF SIMULATION

The second example (i.e., ex02.s) includes a dependency on register t1.

START OF SIMULATION

CDII	Cycles ===>	1	2	3	4	5	6	7	8	9
	\$t1,\$s0,\$s0		2	3	4	5	O	1	O	9
		IL	•	•	•	•	•	•	•	•
	\$t2,\$s0,42	•	•	•	•	•	•	•	•	•
aaa	\$t4,\$t1,70	•	•	•	•	•	•	•	•	•
anıı	a .		•			_		_		_
	Cycles ===>		2	3	4	5	6	7	8	9
	\$t1,\$s0,\$s0	1F	ID	•	•	•	•	•	•	•
	\$t2,\$s0,42	•	IF	•	•	•	•	•	•	•
add	\$t4,\$t1,70	•	•	•	•	•	•	•	•	•
	Cycles ===>		2	3	4	5	6	7	8	9
	\$t1,\$s0,\$s0	IF	ID	EX	•	•	•	•	•	•
	\$t2,\$s0,42	•	IF	ID	•	•	•		•	•
add	\$t4,\$t1,70		•	IF	•	•	•		•	
	A ~			4 D	4	. T	T	T _ 1		
	Cycles	signi	men	t Pro	ηесι	5EX8	tm r	1elp	8	9
	\$t1,\$s0,\$s0	IF	ID	EX	MEM	•	•	. •	•	•
add	\$t2,\$s0,42	•	IF	ID	EX	•	•	•		•
add	\$t4,\$t1,70	htt	na.//	FOIT	^{ID} Od	or co	im		•	
		1111	h2. //	pow	Cou	er.co	JIII			
CPU	Cycles ===>	1	2	3	4	5	6	7	8	9
add	\$t1,\$s0,\$s0	TF	ID	EX	MEM	WB				
aaa	Ψ01,Ψ00,Ψ00		, .		11111	WD	•	•	•	•
	\$t2,\$s0,42	. A d		/ <u> </u>	ist n	MEN X/C	odei	· ~.		
		Ad	led W	Ch	at p	ÖWC	odeı	· •.		
add nop		Ad		Ch	at p	WC ID	odeı	•	· ·	
add nop	\$t2,\$s0,42	Ad		EC	at p	ÿ WC	odeı		· · ·	•
add nop add	\$t2,\$s0,42	Ad		EC	at p	ÿ WC	ode1			
add nop add CPU	\$t2,\$s0,42 \$t4,\$t1,70	Ad	ld W	ECh	at p	ÖWC		•		
add nop add CPU add	<pre>\$t2,\$s0,42 \$t4,\$t1,70 Cycles ===></pre>	Ad	W W 2	F IF 3	Eat p ID 4	TID 5		•		9
add nop add CPU add add	<pre>\$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0</pre>	Ad	Lef W	IF 3 EX	Ext p ID 4 MEM	ID 5 WB	6	•		9
add nop add CPU add add nop	<pre>\$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0</pre>	Ad	Lef W	IF 3 EX ID	Ext p ID 4 MEM EX	ID 5 WB MEM	6 WB	•		9
add nop add CPU add add nop	<pre>\$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42</pre>	Ad	Lef W	IF 3 EX ID IF	In p ID 4 MEM EX ID	ID 5 WB MEM *	6 WB	•	8	
add nop add CPU add add nop add	<pre>\$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42 \$t4,\$t1,70</pre>	Ad	Lef W	IF 3 EX ID IF	In p ID 4 MEM EX ID	ID 5 WB MEM *	6 WB	•		
add nop add CPU add add nop add	<pre>\$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42 \$t4,\$t1,70 Cycles ===></pre>	. Ad	LE W	IF 3 EX ID IF IF IF 3	ID P ID 4 MEM EX ID ID 4	ID 5 WB MEM * ID	6 WB * EX			
add nop add CPU add add nop add	<pre>\$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0</pre>	. Ad	2 ID IF . 2 ID	IF 3 EX ID IF IF 3 EX	ID 4 MEM EX ID ID 4 MEM	ID 5 WB MEM * ID 5 WB	6 WB * EX			
add nop add CPU add nop add CPU add	<pre>\$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42 \$t4,\$t1,70 Cycles ===></pre>	. Ad	LE W	IF IF 3 EX ID IF IF 3 EX ID	ID 4 MEM EX ID ID 4 MEM EX	ID 5 WB MEM * ID 5 WB MEM	6 WB * EX 6 WB	· · · · · · · · · · · · · · · · · · ·		
add nop add CPU add add nop add CPU add nop	<pre>\$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42</pre>	. Ad	2 ID IF . 2 ID	IF IF 3 EX ID IF IF 3 EX ID IF	ID 4 MEM EX ID ID 4 MEM EX ID ID	ID 5 WB MEM * ID 5 WB MEM * MEM * MEM * * * * * * * * * * * *	6 WB * EX 6 WB	· · · · · · · · · · · · · · · · · · ·		
add nop add CPU add add nop add CPU add nop	<pre>\$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0</pre>	. Ad	2 ID IF . 2 ID	IF IF 3 EX ID IF IF 3 EX ID	ID 4 MEM EX ID ID 4 MEM EX	ID 5 WB MEM * ID 5 WB MEM	6 WB * EX 6 WB	· · · · · · · · · · · · · · · · · · ·		
add nop add CPU add add nop add CPU add add nop	<pre>\$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42 \$t4,\$t1,70</pre>	. Ad	LE W	IF IF 3 EX ID IF 3 EX ID IF	ID 4 MEM EX ID ID 4 MEM EX ID ID	ID 5 WB MEM * ID 5 WB MEM 5 WB MEM TID 10	6 WB * EX 6 WB *	7		
add nop add CPU add add nop add CPU add add nop add	<pre>\$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> Cycles ===></pre>	. Ad	LE W	IF IF 3 EX ID IF IF IF 3 EX ID IF IF IF 3	ID P ID 4 MEM EX ID ID 4 MEM EX ID ID 4 MEM EX ID ID 4 4 MEM EX ID ID	ID 5 WB MEM * ID 5 WB MEM 5 WB MEM * ID 5	6 WB * EX 6 WB	· · · · · · · · · · · · · · · · · · ·		
add nop add add nop add CPU add add nop add CPU add add nop add	<pre>\$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0</pre>	. Ad	E W	IF IF 3 EX ID IF 3 EX ID IF IF 3 EX ID IF IF IF	ID 4 MEM EX ID ID 4 MEM EX ID ID 4 MEM EX ID ID 4 MEM EX ID ID 4 MEM EX ID ID	ID 5 WB MEM * ID	6 . WB * EX 6 . WB * EX	7		
add nop add CPU add add nop add CPU add add nop add	<pre>\$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> Cycles ===></pre>	. Ad	LE W	IF IF 3 EX ID IF IF 3 EX ID IF IF IF IF IF IF	ID 4 MEM EX ID ID 4 MEM EX ID ID 4 MEM EX ID ID 4 MEM EX ID ID	ID 5 WB MEM * ID	6 . WB * EX 6 . WB WB	7		
add nop add CPU add add nop add CPU add add nop add CPU add add nop	<pre>\$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0 \$t2,\$s0,42 \$t4,\$t1,70 Cycles ===> \$t1,\$s0,\$s0</pre>	. Ad	E W	IF IF 3 EX ID IF 3 EX ID IF IF 3 EX ID IF IF IF	ID 4 MEM EX ID ID 4 MEM EX ID ID 4 MEM EX ID ID 4 MEM EX ID ID 4 MEM EX ID ID	ID 5 WB MEM * ID	6 . WB * EX 6 . WB * EX	7		

END OF SIMULATION

The third example (i.e., ex03.s) includes two dependencies on register \$t2.

START OF SIMULATION

CPU Cycles ===>	1	2	3	4	5	6	7	8	9
lw \$t2,20(\$a0)	IF			•	•		•		
and \$t4,\$t2,\$t5					•		•		
or \$t8,\$t2,\$t6	•			•	•		•		
CPU Cycles ===>	1	2	3	4	5	6	7	8	9
lw \$t2,20(\$a0)	IF	ID		•		•	•		
and \$t4,\$t2,\$t5	•	IF			•	•	•		
or \$t8,\$t2,\$t6	•			•	•		•		
CPU Cycles ===>	1	2	3	4	5	6	7	8	9
lw \$t2,20(\$a0)	IF	ID	EX	•	•	•	•	•	•
and \$t4,\$t2,\$t5	•	IF	ID	•	•	•	•	•	•
or \$t8,\$t2,\$t6	•	•	IF	•			† 1	•	•
$\mathbf{A}\mathbf{S}$	sign:	men	t Pro).1ec1	LEX	am t	J elp	_	_
CPU Cycles ===>	1 0	_	•	_	5	6	7	8	9
lw \$t2,20(\$a0)	IF	ID	EX	MEM	•	•	•	•	•
nop	htt	IF		*	er.co	im	•	•	•
nop	· 1111	ps: //				JIII	•	•	•
and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6	•	11	†D	ID IF	•	•	•	•	•
			IF	1r				_	
οι ψου,ψοΣ,ψου					•	•	•	•	
	1 Ac	ld W			OWC	ode	[7	8	9
CPU Cycles ===> lw \$t2,20(\$a0)	$^{1}_{1F}$ Ac	ld W			QWC	ode	C ₇	8	9
CPU Cycles ===>			eCh	at p		ode	C ₇	8	9 .
CPU Cycles ===> lw \$t2,20(\$a0)		ID	eCh	nat p	WB	ode :	C ₇	8	9 .
CPU Cycles ===> lw \$t2,20(\$a0) nop	IF	ID IF	eCh	nat p	WB *	ode	[7	8	9
CPU Cycles ===> lw \$t2,20(\$a0) nop nop	IF	ID IF IF	eCh	nat p MEM * *	WB * *	ode : :	C ₇	8	9
CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6	IF	ID IF IF IF	EX ID ID ID ID	nat p MEM * ID	WB * ID	ode		8	9
CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6 CPU Cycles ===>	IF	ID IF IF	EX ID ID ID IF	nat p MEM * ID IF	WB * ID	ode1	7	8	9 9
CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6	IF	ID IF IF IF C ID	EX ID ID ID IF	MEM * * ID IF	WB * * ID IF	· · · · ·	· · · · ·		
CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6 CPU Cycles ===>	IF	ID IF IF IF 2 ID IF	EX ID ID ID IF 3 EX ID	nat p MEM * ID IF	WB * ID IF	· · · · ·	· · · · ·		
CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6 CPU Cycles ===> lw \$t2,20(\$a0) nop nop	IF	ID IF IF IF . 2 ID IF IF	EX ID ID IF 3 EX ID ID	MEM * * ID IF 4 MEM * * *	WB * * ID IF 5 WB * *	6 . *	· · · · ·		
CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6 CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5	IF	ID IF IF IF 2 ID IF	EX ID ID IF 3 EX ID ID IT	MEM * ID IF 4 MEM * ID IT	WB * ID IF 5 WB * ID	6 . * * EX	· · · · ·		
CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6 CPU Cycles ===> lw \$t2,20(\$a0) nop nop	IF	ID IF IF IF . 2 ID IF IF	EX ID ID IF 3 EX ID ID	MEM * * ID IF 4 MEM * * *	WB * * ID IF 5 WB * *	6 . *	· · · · ·		
CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6 CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6	IF IF	ID IF IF . 2 ID IF IF IF .	EX ID ID IF 3 EX ID	MEM * * ID IF 4 MEM * * ID IF	WB * ID IF 5 WB * ID IF	6 * * EX	· · · · · · · · · · · · · · · · · · ·		9
CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6 CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6 CPU Cycles ===>	IF	ID IF IF IF . 2 ID IF IF IF .	EX ID ID IF 3 EX ID ID IF 3 EX ID ID ID IF 3	MEM * ID IF 4 MEM * ID IF 4 MEM 4	WB * ID IF 5 WB * ID IF 5 5 5 5 5 5	6 . * EX ID	· · · · ·		
CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6 CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6 CPU Cycles ===> lw \$t2,20(\$a0)	IF IF	ID IF IF IF 2 ID IF IF IF 2 ID	EX ID ID IF S EX ID ID IF 3 EX ID ID ID IF 3 EX	MEM * * ID IF 4 MEM * * ID IF	WB * ID IF 5 WB * ID IF	6 * * EX	· · · · · · · · · · · · · · · · · · ·		9
CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6 CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6 CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6	IF	ID IF IF IF 2 ID IF IF IF IF IF IF	EX ID ID IF 3 EX ID	MEM * * ID IF 4 MEM * ID IF 4 MEM 4 MEM	WB * ID IF 5 WB * ID IF 5 WB WB	6 . * EX ID	· · · · · · · · · · · · · · · · · · ·		9
CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6 CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6 CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6	IF	ID IF IF IF 2 ID IF IF IF 2 ID IF IF	EX ID ID IF 3 EX ID ID ID IF 3 EX ID ID ID IF	MEM * ID IF 4 MEM * ID IF 4 MEM * MEM *	WB * ID IF 5 WB * ID IF 5 WB * * * * * * * * * * * * * * * * * *		· · · · · · · · · · · · · · · · · · ·		9
CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6 CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6 CPU Cycles ===> lw \$t2,20(\$a0) nop nop and \$t4,\$t2,\$t5 or \$t8,\$t2,\$t6	IF	ID IF IF IF 2 ID IF IF IF IF IF IF	EX ID ID IF 3 EX ID	MEM * ID IF 4 MEM * ID IF 4 MEM 4 MEM * ID IF	WB * ID IF 5 WB * ID IF 5 WB * WB *		· · · · · · · · · · · · · · · · · · ·		9

CPU Cycles ===>	1	2	3	4	5	6	7	8	9
lw \$t2,20(\$a0)	IF	ID	EX	MEM	WB	•		•	
nop		IF	ID	*	*	*			
nop		IF	ID	*	*	*			
and \$t4,\$t2,\$t5		IF	ID	ID	ID	EX	MEM	WB	
or \$t8,\$t2,\$t6			IF	IF	IF	ID	EX	MEM	
CPU Cycles ===>	1	2	3	4	5	6	7	8	9
lw \$t2,20(\$a0)	IF	ID	EX	MEM	WB		•		
nop	•	IF	ID	*	*	*		•	
nop	•	IF	ID	*	*	*			
and \$t4,\$t2,\$t5	•	IF	ID	ID	ID	EX	MEM	WB	
or \$t8,\$t2,\$t6		•	IF	IF	IF	ID	EX	MEM	WB

END OF SIMULATION

Assumptions

Given the cassignment, Projecthe ExamuHelp

- Assume all input files are valid.
- Assume the length of present.

Error Checking Add WeChat powcoder

Given the complexity of this assignment, you can assume that the input file is valid. Be sure to verify that you have the correct number of arguments by checking argc; display an error message if argument(s) are missing.

In general, if an error occurs, use either perror() or fprintf(stderr, "..."), depending on whether the global errno is set.

And be sure to return either EXIT_SUCCESS or EXIT_FAILURE upon program termination.

Submission Instructions

Before you submit your code, be sure that you have clearly commented your code (this should not be an after-thought). Further, your code should have a clear and logical organization.

To submit your assignment (and also perform final testing of your code), please use Submitty. Note that the test cases for this assignment will be available on Submitty a minimum of three days before the due date and will include hidden test cases.

Also as a reminder, your code **must** successfully execute on Submitty to obtain credit for this assignment.