<u>Dashboard</u> / <u>My courses</u> / <u>COSC3360SP202502</u> / <u>PRACTICE EXAMS</u> / <u>Exam 1 - Practice</u>

Status	Finished			
Started	Thursday, 6 February 2025, 12:02 AM			
Completed	Completed Thursday, 6 February 2025, 12:13 AM			
Duration	10 mins 34 secs			
Grade	100.00 out of 100.00			
Information				

True and False questions (on Exam 1 you will get more questions of this type)

Question 1
Correct
Mark 10.00 out of 10.00
In a priority scheme to handle multiple interrupts, the system disables the interrupts while an interrupt is being processed.
Select one:
○ True
® False ⊘
The correct answer is 'False'.
Information

Simple Choice questions (on Exam 1 you will get more questions of this type)

Question 2	
Correct	
Mark 10.00 out of 10.00	

Select the element that is not part of the process control block:

a. PIDb. Priority

oc. State

 $\ \ \,$ d. Program code $\ \ \,$

e. None of the above

The correct answer is: Program code

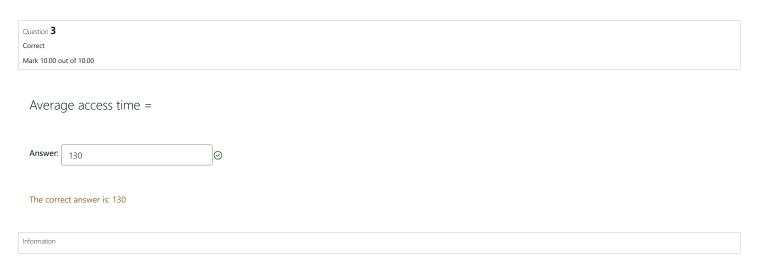
Calculate the average acce	ess time to read a v	word from memory	v for the following two	-level memory system.

Level 1 memory access time = 50 ms

Level 2 memory access time = 400 ms

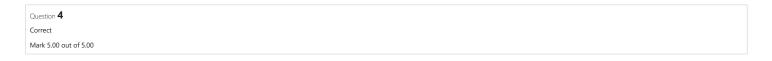
Failure ratio = 20%

Time to find a word in any level of the memory (0 ms). (15 points).



Calculate the following parameters of a hypothetical computer system with these features:

- a) Octal notation;
- b) IR = OPCode + Mem Addr;
- c) IR = 12 bits;
- d) PC = 3 octal digits; and
- e) Mem word size = Data (unsigned integer) = IR. (15 points)



Number of different OPCodes:



The correct answer is: 8

Question 5
Correct
Mark 5.00 out of 5.00
Mem size in bits:
Well Size III olds.
Answer: 6144
The correct answer is: 6144
Question 6
Correct
Mark 5.00 out of 5.00
Mem range: 000 - 777
000 - F77 000 - FFF 0000 - 7777
The correct answer is:
Mem range: [000 - 777]
Question 7
Correct
Mark 5.00 out of 5.00
Data range: 0000 - 7777
0000 - FFFF 0000 - 777 0 -7
The correct answer is:
Data range: [0000 - 7777]

```
Question 8
Correct
Mark 30.00 out of 30.00
```

Complete the C++ program below to generate the following process tree:

```
UID
     PID PPID
                 C STIME
                           TTY
                                         TIME CMD
                                      0:00.01 /bin/bash -l
501 18282 18281
                 0 6:21AM ttys000
501 18360 18359
                 0 6:26AM ttys001
                                      0:00.04 -bash
501 18419 18360
                 0 6:29AM ttys001
                                      0:00.00 ./PQ1
                 0 6:29AM ttys001
501 18420 18419
                                      0:00.00 ./PQ1
                 0 6:29AM ttys001
501 18421 18419
                                      0:00.00 ./PQ1
                 0 6:29AM ttys001
                                      0:00.00 ./PQ1
501 18422 18419
                 0 6:29AM ttys001
501 18423 18421
                                      0:00.00 ./PQ1
                                      0:00.01 -bash
501 18413 18412
                0 6:29AM ttys002
```

Notes:

- 1. PQ1 is the name of the process.
- 2. Use wait and _exit when needed to guarantee the following output:

```
I am the parent process
I am the child process 0
I am the child process 1
I am a grandchild process from child process 1
I am the child process 2
```

Answer: (penalty regime: 0 %)

```
Reset answer
     #include <iostream>
      #include <unistd.h>
     #include <sys/wait.h>
  4
  5
     int main()
  6
  7
          int pid:
           std::cout << "I am the parent process" << std::endl;</pre>
  8
           for(int i = 0; i < 3; i++){
               if((pid=fork())==0){
 10 •
                    std::cout << "I am the child process " << i << std::endl;
 12 🔻
                    if(i==1){
 13 •
                        if((pid=fork())==0){
 14
                             std::cout << "I am a grandchild process from child process " << i << std::endl;</pre>
 15
                             _exit(0);
 16
                        wait(nullptr);
 17
 18
 19
                    _exit(0);
 21
 22
               wait(nullptr);
 23
 24
       // std::cout << "I am the parent process" << std::endl; // std::cout << "I am the child process " << /*variable identifier*/ << std::endl;
 25
 26
       // std::cout << "I am a grandchild process from child process " << /*variable identifier*/ << std::endl;
 28
          return 0;
```

	Expected	Got	
0	I am the parent process	I am the parent process	0
	I am the child process 0 I am the child process 1	I am the child process 0 I am the child process 1	
	I am a grandchild process from child process 1 I am the child process 2	I am a grandchild process from child process 1 I am the child process 2	

Passed all tests! 📀

► Show/hide question author's solution (Cpp)

Correct
Marks for this submission: 30.00/30.00.

```
Question 9
Correct
Mark 20.00 out of 20.00
```

The multithreaded calculator.

You must complete the program below (using POSIX threads) that creates four threads to calculate the addition, subtraction, multiplication, and division of a set of numbers read from STDIN.

The input of the program will be four pair of integer values with the following format:

```
1 2
2 3
4 5
6 7
```

Where the first line has the values to execute the addition, the second line has the values to execute the subtraction, the third line has the values to execute the multiplication, and the last line has the values to execute the division. For the division operator, if the denominator is equal to zero, the result of the operation is zero.

Given the previous input, the expected output is:

```
1 + 2 = 3.00
2 - 3 = -1.00
4 * 5 = 20.00
6 / 7 = 0.86
```

Notes:

- 1. Use the comments in the provided template file to complete your solution.
- 2. Not using POSIX threads will translate into a penalty of 100%.
- 3. You can always assume that the input will be valid.

For example:

Input	Result
10 20	10 + 20 = 30.00
5 2	5 - 2 = 3.00
3 2	3 * 2 = 6.00
1 100	1 / 100 = 0.01

Answer: (penalty regime: 0 %)

```
Reset answer
                      pos_ptr->result = pos_ptr->val1 * pos_ptr->val2;
28
                     break:
             case 3: // Division
29
30
                     if(pos_ptr->val2 == 0)
31
                         pos_ptr->result = 0;
32
 33
                         pos_ptr->result = (1.0*pos_ptr->val1)/pos_ptr->val2;
 34
 35
36
         return NULL;
37
38
     int main()
39
40
41
         static struct operation operations[NOPER];
         pthread_t tid[NOPER];
 42
 43
44
         for(int i=0;i<NOPER;i++)</pre>
45
46
             operations[i].op = i;
             std::cin >> operations[i].val1;
47
             std::cin >> operations[i].val2;
48
             if(pthread_create(&tid[i], NULL, calculator, &operations[i]))/* Call pthread_create */
49
50
                  fprintf(stderr, "Error creating thread\n");
 51
 52
                 return 1;
 53
54
55
56
         // Wait for the other threads to finish.
57
58
         // Call pthread_join here
         for(int x = 0; x < NOPER; x++)
59
             pthread_join(tid[x], nullptr);
 60
 61
 62
         for (int i = 0; i < NOPER; i++)</pre>
 63
 64
             switch(operations[i].op)
```

```
case 0: std::cout << operations[i].val1 << " + " << operations[i].val2 << " = " << std::fixed << std::setprecision(2) << operations[i].result break;

case 1: std::cout << operations[i].val1 << " - " << operations[i].val2 << " = " << std::fixed << std::setprecision(2) << operations[i].result break;

case 2: std::cout << operations[i].val1 << " * " << operations[i].val2 << " = " << std::fixed << std::setprecision(2) << operations[i].result break;

case 3: std::cout << operations[i].val1 << " / " << operations[i].val2 << " = " << std::fixed << std::setprecision(2) << operations[i].result break;

case 3: std::cout << operations[i].val1 << " / " << operations[i].val2 << " = " << std::fixed << std::setprecision(2) << operations[i].result break;

}
return 0;

return 0;
```

	Input	Expected	Got	
0	10 20	10 + 20 = 30.00	10 + 20 = 30.00	0
	5 2	5 - 2 = 3.00	5 - 2 = 3.00	
	3 2	3 * 2 = 6.00	3 * 2 = 6.00	
	1 100	1 / 100 = 0.01	1 / 100 = 0.01	
0	1 2	1 + 2 = 3.00	1 + 2 = 3.00	0
	100 200	100 - 200 = -100.00	100 - 200 = -100.00	
	3 7	3 * 7 = 21.00	3 * 7 = 21.00	
	50 3	50 / 3 = 16.67	50 / 3 = 16.67	
0	45 3	45 + 3 = 48.00	45 + 3 = 48.00	0
	33 2	33 - 2 = 31.00	33 - 2 = 31.00	
	10 9	10 * 9 = 90.00	10 * 9 = 90.00	
	100 0	100 / 0 = 0.00	100 / 0 = 0.00	

Passed all tests! 🔗

► Show/hide question author's solution (Cpp)

Correct

Marks for this submission: 20.00/20.00.

→ Programming Assignment 1

Jump to...

Exam 2 - Practice

/