NOTE: The problems have been carefully checked before they are given to you. However, if there are questions which seem unclear or not well-defined to you, don't worry. You can make your own assumptions in such cases. Please state clearly in the answer sheet what's the problems and the assumptions you made. The grading is flexible.

1. Performance Analysis (25%)

Speedup is the measure of how a computer performs after some enhancement relative to how it performed previously. The speedup is defined as:

$$Speedup = \frac{\text{Performance after improvement}}{\text{Performance before improvement}} = \frac{\text{Execution time before improvement}}{\text{Execution time after improvement}}$$

where

$$\text{Execution time after improvement} = \frac{\text{Execution time affected by improvement}}{\text{Amount of improvement}} + \text{Execution time unaffected}$$

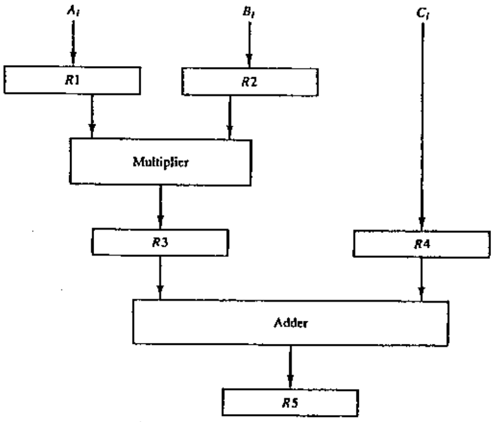
1.1 (20%) Suppose we enhance a computer to make all floating-point instructions run five times faster. Let's look at how speedup behaves when we incorporate the faster floating-point hardware into the computer. If the execution time of some program before the floating-point enhancement is 10 seconds, what will the speedup be if 25%, 50%, 75% or 100% of the 10 seconds is spent executing floating-point instructions, respectively? Please show your calculation in your answer sheet, copy the following table to your answer sheet and fill it with the answers.

Fraction of executed floating-point instructions in the program	Speedup
25%	
50%	
75%	
100%	

1.2 (5%) We are looking for a program to demonstrate that the new floating-point unit described in Question 1.1 gives a speedup of 3. If the execution time of one such program is 100 seconds with the old floating-point hardware, how much percentage of the original execution time should be executing floating-point instructions in order to show an over speedup of 3?

2. Pipeline Structure (25%)

The pipeline of the following figure has the propagation times: 45 ns for the operands to be read from memory into registers R1 and R2, 50 ns for the signal to propagate through the multiplier, 5 ns for the transfer into R3, and 20 ns to add the two numbers into R5.



2.1 (5%) What is the minimum clock cycle time that can be used?

2.2 (5%) A non-pipeline system can perform the same operation by removing R3 and R4. How long will it take to multiply and add the operands without using the pipeline?

2.3 (10%) Calculate the speedup of the pipeline (with respect to the non-pipeline system) for 10 tasks and 100 tasks, respectively.

2.4 (5%) What is the maximum speedup that can be achieved?

3. Explain each of the following terms. (10%)

(a) NP

(b) NP-complete

(c) greedy method

(d) lower bound of a problem

(e) prune and search

4. Define the *Fibonacci binary tree* of order n as follows: If $n = 0$ or $n = 1$, then the tree consists of a single node. If $n > 1$, the tree consists of a root, with the Fibonacci tree of order $n - 1$ as the left subtree and the Fibonacci tree of order $n - 2$ as the right subtree.

(a) Draw a Fibonacci tree of order 4. (3%)

(b) How many leaves are there in the Fibonacci tree of order n, $n > 1$? (4%)

(c) What is the depth of the Fibonacci tree of order n, $n > 1$? Assume that the depth of the tree consisting of a single node is 1. (4%)

(d) What is a height-balanced binary tree? Is a Fibonacci tree a height-balanced binary tree? Why? (4%)

5. There is a recursive C function to find the maximum of n elements stored in an array:

```
max(int a[ ], int n)
/* The data elements are stored in a[1],a[2],...,a[n] */
/* n: number of elements in array a[ ] */
{
  printf("ENTERING");
  if(n==1)
    return(a[1]);
  else
    if (a[n]>=max(a,n-1))
      return(a[n]);
    else
      return(max(a,n-1))
}
```

In the main program, we have a call $max(a, m)$, $m \geq 1$, to the above function.

(a) How many times the message ENTERING is printed at least? What situation will cause the least number of message ENTERING to be printed? (5%)

(b) How many times the message ENTERING is printed at most? What situation will cause the most number of message ENTERING to be printed? (5%)

(c) How many times the message ENTERING is printed in average? It is assumed that the data are in a uniform distribution. And, in this problem, you need only to write down the recurrence formula, need not derive the formula. (5%)

6. Let $P = (p_1, p_2, \cdots, p_n)$ be a permutation of $\{1, 2, \cdots, n\}$. g_i, $2 \leq i \leq n$, is a permutation operator that $g_i(P) = (p_i, p_{i-1}, \cdots, p_3, p_2, p_1, p_{i+1}, p_{i+2}, \cdots, p_n)$. Consider $n = 5$. For example, $g_4(51423) = (24153)$ and $g_3(12345) = (32145)$.

(a) Given a permutation (23514), what sequence of permutation operators can be applied to get the permutation (12345)? (3%)

(b) (12345) is called the identity permutation. For any given permutation, design a general algorithm for deciding how to apply the permutation operators to transform the given permutation to the identity permutation. How many permutation operators are required for any value of n in the worst case? (7%)