※ 注意：請於試卷上「非選擇題作答區」依序作答，並應註明作答之大題及小題題號。

1. (20%) The following table describes the encoding mechanism of MIME Base64. This encoding can transform an arbitrary sequence of octets to a format that can be expressed in short lines of 7-bit characters. The current version of MIME Base64 is based on RFC-1421. The 64 characters of Base64 (char table) are defined as following:

Index	0 1 2 3 4 5 6 0123456789012345678901234567890123456789012345678901234567890123
Base64 Char	ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/

The following figure describes the encoding process of the MIME Base64.

Hex pattern	0x4D	0x61	0x6E
Bit pattern	0 1 0 0 1 1 0 1	0 1 1 0 0 0 0 1	0 1 1 0 1 1 1 0
Index	19	22	5	46
Base64-Encoded	**T**	**W**	**F**	**u**
ASCII value	0x54	0x57	0x46	0x75

8-bit to 6-bit transform
Base64 char table lookup
Store 8-bit ASCII value

a. (15%) Please write a function or a method (in C/C++ or Java) that can **convert 3** bytes of general binary data into 4 Base64 characters (Hints: You can use the bit and shift operators (| & ^ ~ >> <<) with appropriate masks)

b. (5%) When you try to transfer a photo to your friends in email, the binary data of the photo will be encoded in MIME Base64. If your photo is 123,456 bytes, how large will it become after the photo is attached in your email?

2. (20%) The following table lists the performance measurement of 3 computer platforms with the same benchmark program. After benchmarking with the program, we know Computer **B** performs as fast as Computer **A**, and Computer **C** performs faster than Computer **A**.

Benchmark platform	Computer A	Computer B	Computer C
Instruction counts	100 millions	100 millions	180 millions
Clock rate	3.6 Ghz	2.8Ghz	?
Cycles per Instruction	1.8	?	1.2

a. (5%) What is the value of **cycles per instruction** of Computer B?
b. (5%) Computer A and B use the same CPU architecture with different **clock rate** and different **cache size**, which computer has larger cache? Explain your answer.
c. (5%) What is the requirement for the **clock rate** of Computer **C**?
d. (5%) Why the same benchmark program compiled for Computer **C** is larger than Computer **A**? If there is only one RISC machine among Computer **A B C**, which one looks like a **RISC** machine? Explain your answer according to the design philosophy RISC.

3. (10%) In C, we can call the swap action of two numbers as the following code fragment.

```
int a = 10, b = 20; Swap(a, b);
```

There are 3 codes fragments that try to implement the above `Swap()` functionality.

Code A:
```
void Swap(int a, void b)
{
    a ^= b;
    b ^= a;
    a ^= b;
}
```

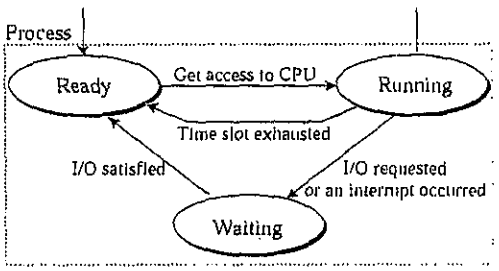
Code B:
```
void Swap(int *a, void *b)
{
    *a ^= *b;
    *b ^= *a;
    *a ^= *b;
}
```

Code C:
```
#define Swap(a, b) \
    (a^= b, b ^= a, a ^= b;)
```

a. (3%) Is the Code **A** workable? Explain your answer.
b. (4%) Prove that exclusive-OR (^) operation can swap two integers.
c. (3%) Compare the advantages and disadvantages of Code **B** and **C**.

見背面

4. (10%) Starvation problem.
   a. Left figure is the state diagram of process in system. Give an example and explain how **starvation** can happen in **Ready** state? Also give an example and explain how it happens in **waiting** state?
   b. And propose your solutions for the avoidance of starvation in each state?



5. (20%) Iterative and Recursive Programming.
   The Ackermann function is defined as:

$$Ack(m,n)=\begin{cases} n+1 & \text{if } m=0, \\ Ack(m-1,1) & \text{if } m>0 \text{ and } n=0, \\ Ack(m-1,Ack(m,n-1)) & \text{if } m>0 \text{ and } n>0, \end{cases}$$

   a. Please use **recursive** approach to write a function (in C/C++ or Java) that can calculate the $Ack(m,n)$, where $0 \le m \le 3$ and $0 \le n \le 20$.
   b. Please use **two dimensional array** data structure and **iterative/loop** approach to write a function (in C/C++ or Java) that can calculate the the $Ack(m,n)$, where $0 \le m \le 3$ and $0 \le n \le 20$.

6. (20%)
   a. (14%) Write an **insertion sort** program (in C/C++ or Java) that can sort an array of integers in non-decreasing order?
   b. (3%) What's the best case for the **insertion sort** program? Give an example array of 8 integers to explain the best case.
   c. (3%) What's the time complexity of this best case for ***n*** integers array?

試題隨卷繳回