quiz1. Code=1 Digipen login:__________________________

1. **Problem** (8 pts):

   One of the three most useful bit manipulating operations is setting a specific bit to 1.

   write code that sets n'th bit of integer i to 1, points are awarded for correctness and compactness:

2. **Problem** (2 * 3 pts):

   Let x be a `char` (8 bits) and another `char` $mask = 7 = 00000111_2 = 2^2 + 2^1 + 2^0$. That is $mask$ has only 3 bits set to 1 – at positions 0,1,2. Find English sentence A. . . N (all referring to x) which is equivalent to the given C-statement.

   Note: the meaning of "is true" – when the statement is used in an if-statement, the *then* part will be executed, which is equivalent to arithmetical "not equal to zero".

A) At least one of the bits 0,1,2 is 0
B) Bits 3,4,5,6,7 are all 0's
C) Bits 3,4,5,6,7 are all 1's
D) At least one of the bits 3,4,5,6,7 is 0
E) At least one of the bits 3,4,5,6,7 is 1
F) Always false
G) Bits 0,1,2 are all 1's
H) Bits 0,1,2 are all 0's
I) x equals to 7
J) Always true
K) At least one of the bits 0,1,2 is 1
L) x equals to 0
	2-1.______
	`( ~(x & mask) )        is true`
	2-2.______
	`( ~x & mask )          is true`

3. **Problem** (4 * 2 pts):

   Convert C declaration into English

   A) an array of 5 pointers to pointers to int

   B) a pointer to a pointer to a function that returns an int

   C) a function that takes an int and returns a pointer to a pointer to an int

   D) a pointer to an array of 5 pointers to int

   E) an array of 5 pointers to functions taking an int and returning an int

   F) a pointer to a function that takes an int and returns an int

   G) a pointer to an array of 5 pointers to functions that take nothing and return an int

   H) legal, but not on the list

   I) a pointer to an array of 5 pointers to functions taking an int and returning an int

   J) a pointer to a function that takes an int and returns a pointer to an int

   K) an array of 5 pointers to functions taking an int and returning a pointer to int

   L) a pointer to a function taking int and returning an array of 5 ints

   M) a pointer to a pointer to an array of 5 int

   N) a pointer to a function taking int and returning a pointer to an array of 5 ints

   O) illegal declaration

   3-1.______ `int (**foo)[5];`

   3-2.______ `int ((*foo)(int))[5];`

   3-3.______ `int (*(*foo)[5])(int)`

   3-4.______ `int * (*foo[5])(int)`

4. **Problem** (2 * 3 pts):

   Choose corresponding C-style declaration for each of the English statements below

   A) `int *(foo(int))[5]`

   B) `int foo(int)*[5]`

   C) `int foo(int)[5]`

   D) `int *(*foo(int))[5]`

   E) `int *(foo[5])(int**[5])`

   F) `int (*foo[5])[5](int*(*))`

   G) `int (*foo[5])((*)[5]int*)`

   H) `int int*(*foo[5])((*)[5])`

   I) `illegal declaration`

   J) `legal, but not on the list`

   K) `int ([5]*foo(int))`

   L) `int (*foo(int))[5]`

   M) `(*foo(int))[5]int`

   N) `int [5](*foo(int))`

   O) `int (*foo[5])(int*(*)[5])`

   P) `int (*foo[5])(int*(*))[5]`

   4-1.______ foo is a function taking taking int and returning a pointer to an array of 5 ints

   4-2.______ foo is an array of 5 pointers to functions taking (a pointer to a array of 5 pointers to int) and returning an int

5. **Problem** (6 * 1 pts):

   For each of the following expressions determine whether it's legal.
   For legal expression write down its value.
   Assume non-cumulative execution, i.e. all modifications from previous lines ARE LOST.

```
int c[]={5,7,10,3,1};
int *pc=c;
// assume pc = 1000; integer is 4 bytes
```

A) illegal B) 3 C) 4 D) 5 E) 10 F) 1002 G) 1003 H) 1004 I) 1000 J) 1001 K) 7 L) 6 M) 1012 N) 8	5-1.______ `*pc;` 5-2.______ `*pc++;` 5-3.______ `*++pc;` 5-4.______ `(*++pc)++;` 5-5.______ `pc+c[3];` 5-6.______ `*pc+c[3];`

6. **Problem** (7 * 1 pts):

   Which of the following assignments are legal? Assume Foo is a well-defined struct. Unless declaration is provided in a question, assume

```
Foo f;
const Foo cf;
Foo* p_f;
const Foo* p_cf;
Foo* const cp_f;
```

A) illegal B) legal	6-1.______ `Foo* p_f = &f;` 6-2.______ `Foo* p_f = &cf;` 6-3.______ `Foo * const cp_f = &f;` 6-4.______ `Foo * const cp_f = &cf;` 6-5.______ `cp_f = &f;` 6-6.______ `Foo* p_f = cp_f;` 6-7.______ `cp_f = p_f;`

7. **Problem** (6 pts):

   Implement a `pop_front` function that deletes a node at the front of a singly-linked list. Function return type should be `void`