Section A

Q	A	Explanation
1	B	One may solve this problem by observing that `f(a)` outputs the digits of `a` but with the order reversed.
2	B	(ii) is true because *D* is a subset of *C*, which is a subset of *B*. (i) is false, for instance, when *D* is a proper subset of *A*.
3	A	At the beginning there are six islands as disconnected components. Constructing a bridge removes the number of disconnected components by at most one. Thus at least five bridges are required to connect the six islands. It is simple to see that five bridges indeed suffices.
4	C	This is the minimum spanning tree problem. One may greedily construct a bridge with the minimum cost if it connects two disconnected components, and repeat the above procedure until all islands are connected.
5	C	Program Y fails because it outputs some Pythagorean triples with *c* greater than 100.
6	D	Program Z is the fastest because it takes approximately $N^2/2$ operations. Program Y takes approximately N^2 operations. Program X is the slowest because it takes approximately N^3 operations.
7	*	The answer for Pascal version is B while that for C/C++ version is D. Note that when `a=30000` and `b=40000`, `a*a+b*b` exceeds the long integer range and causes an overflow error. Program X outputs the required triple because after an overflow error, the condition `a*a+b*b=c*c` (or `a*a+b*b==c*c` in C/C++) still holds. For Pascal version, Programs Y and Z fail to output the required triple because at some point of execution, `a*a+b*b` (or `c*c-a*a`) causes an overflow error and gives a negative number, which when passed to `sqrt` causes a runtime error, terminating the program. For C/C++ version, the overflow error also occurs, but `sqrt` of a negative number is `NaN`, and the program continues to run. C/C++ version of Program Y fails to output the required triple because `floor(NaN)` also gives `NaN`, but `NaN != NaN` by convention. C/C++ version of Program Z outputs the required triple because $50000^2 - 30000^2$ is within the long integer range even though an overflow error occurs.
8	D	The answer is easily seen by simple algebraic manipulations.
9	B	Note that (B nand A) = (A nand B) and that (A nand A) = not A. Thus (A nand B) nand (B nand A) = (A nand B) nand (A nand B) = not (A nand B) = not (not (A and B)) = A and B.
10	B	This is just a simple dry running problem.
11	B	In the question, we may swap any two numbers, even if they are not next to each other. Since four numbers in the given list are misplaced, at most three swaps are needed, because we may swap twice to put any two of the misplaced numbers in their correct places, then swap once more to make the list sorted in ascending order. Since we cannot sort the given list by only two swaps (which happens only when the four misplaced numbers form two pairs), the minimum number of swaps needed is three.
12	D	It may happen that one basket contains all 2004 apples, so both statements may be false.
13	A	The ranges of possible values for the four choices are: -99 to 99 for Choice A, -99 to 100 for Choice B, -99 to 0 for Choice C and -98 to 98 for Choice D.
14	C	A counter example to expression A is A=0, B=1. A counter example to expression B is A=-1, B=0. C is always true because the exponentiation function is strictly increasing for real numbers.
15	A	Divisions in general introduce precision errors. Loosely speaking, less divisions introduce less errors.
16	A	Program segment B does not work when `m` < 0. Program segment C does not work when `m` < 0 or `n` < 0.
17	B	Program segment A obviously works. Program segment B also works by simple algebraic manipulations. Note that the result remains correct even if `a+b` causes an overflow error. Program segment C does not work when, for instance, `b` equals 0 or when `a*b` causes an overflow error.
18	C	If `m` is a Hamming number and `m` is divisible by 2, then `m`/2 must be a positive integer. Moreover, `m`/2 cannot have prime factors other than 2, 3 or 5. Therefore statement (i) is true. By the same argument, statement (ii) is also true.
19	B	To apply Binary Search to an array, the array must be sorted so as to give the correct result (ascending order and descending order are both fine). Moreover, to determine whether a number exists in an array, the numbers in the array need not be unique.
20	B	To truncate a number to 2 decimal places, the simplest way is to multiply it by 100, truncate the product and then multiply the result by 0.01 (or divide the result by 100).
21	D	It is easy to see that `H` and `K` will be printed when `n<50` and `n>50` respectively. Therefore `O` will be printed when `n=50`, but `I` never gets printed.
22	C	This problem actually asks you to count the number of pairs `(i, j)` such that `i` and `j` are both positive integers less than or equal to 10 and `i` is a multiple of `j`. By listing all pairs systematically, we know there are 27 such pairs.
23	D	This is just a simple dry running problem.

Q	A	Explanation
24	B	Note that when the function `f` is called, the parameter `a` is passed by value, so the value of `A` in the main program does not change after calling `f`.

Section B

Q	Possible Answers (Not exhaustive)		Explanation
	Pascal Version:	C/C++ Version:	
A	`div 2`	`/2`	The simplest way to reverse a string is to swap the first and the last characters, swap the second and the second last characters, and so on.
B	`len+1-i`	`len-1-i`	
C	`len+1-i`	`len-1-i`	
D	`(n+m-1) div m`	`(n+m-1)/m`	In this question, contestants are asked to compute $\lceil n/m \rceil$, where $\lceil x \rceil$ denotes the smallest integer greater than or equal to x (this is called the *ceiling* of x). Many contestants gave answers similar to the following mathematical expression: $\text{round}(n/m+0.49999)$, which is in fact prone to precision error.
E	A=True,B=True,C=False,D=True		This problem can be solved by exhausting all 16 possibilities.
F	`(i=j) or (i=8-j)`	`i==j || i==8-j`	All ‘`X`’s lie on the two diagonals, therefore an ‘`X`’ should be printed if and only if `i` equals `j` or `i+j` equals 8.
G	`n div 3`	`n/3`	A number n is a power of 3 if and only if its base 3 representation contains a single ‘`1`’ and all their other digits are ‘`0`’s. Therefore we may first keep dividing the number by three until it is not divisible by three, effectively removing all trailing ‘`0`’s, then check whether the remaining digits comprise a single ‘`1`’ only.
H	`n=1`	`n==1`	
I	89		This problem can be solved easily if one computes `f(1)`, `f(2)`, ..., `f(10)` in order.
J	n-1		By comparing two distinct integers, one number is eliminated from the list of candidates for the smallest integer. Therefore starting with n candidates, at least $n-1$ comparisons are needed to eliminate $n-1$ candidates. It is simple to see that $n-1$ comparisons suffices to pick out the smallest integer.
K	`dst`	`dst`	This problem involves simple linked list manipulations.
L	`src`	`src`	
M	10110		This problem can be solved easily if one traces the program carefully, or computes the outputs of `r(1)`, ..., `f(4)` in order.
N	Any string with two or more “`HKOI2003`”s, or any string longer than 255 characters		The program in fact only replaces the first occurrence of “`HKOI2003`” with “`HKOI2004`”. Also, if the input string is longer than 255 characters, the output string will be truncated to 255 characters only.
O	76		Contestants have to modify Dijkstra shortest path algorithm and apply the modified algorithm to compute the required score. Note that greedily moving to an adjacent cell with the smallest number does not work.
P	112		One may count the number of such 5-digit integers which are larger than or equal to 53412, and then subtract this number from the total number of all such 5-digit integers. Alternatively, one may count the number of 5-digit integers of the forms 1????, 2????, 3????, 4????, 51???, 52???, 531??, 532?? and 5341?.