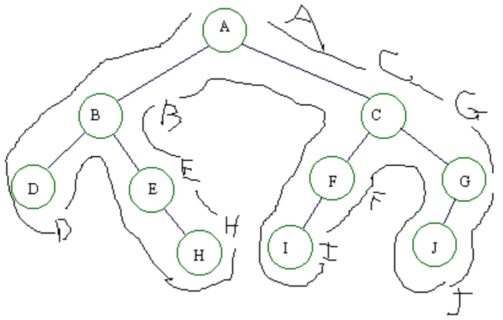
		Solutions
1	B	3B7F Translate each digit 3 = 0011 B = 1011 7 = 0111 F = 1111
2	E	A relation R is reflexive if for all x xRx symmetric if xRy → yRx transitive if xRy and yRz → xRz None of these hold
3	D	if items can't be used more than once then there are 6 ways choose 1st 5 for 2nd, 4 for 3rd and 3 for 4th thus (6)(5)(4)(3)=360
4	A	$R = (ab \mid abb)^* bbab$ The first part means *ab* OR *abb* and the * means repeated as many times as desired. *bbab* means that the expression must end with 1 occurrence of *bbab*. A) does not end it *bbab*.
5	C	recursive function that SUMS all the digits from 8 downto 0 (8)(9)/2 = 36
6	D	Choices B and C can be eliminated immediately since each word must end in a 1 (when you enter D) A is legal, but doesn't handle all the possible alternate paths. One such path is 1011.
7	B	 A B C D E F G H I J A C G B E F D H I J DHEBIFJGCA Draw a path around the tree. every time you see a node for the LAST time, write down its value

		Solutions
8	D	37, 38, 72, 48, 98, 11, 56 Size of storage array is 7 (0..6) then each number must be MOD ed by 7 37 mod 7 = 2 37 in pos 2 38 mod 7 = 3 37 in pos 3 72 mod 7 = 2 * collision in 2 and 3 72 in pos 4 48 mod 7 = 6 48 in pos 6 98 mod 7 = 0 98 in pos 0 11 mod 7 = 4 *collision in 4 11 in pos 5 56 mod 7 = 8 *collision 0 56 in pos 1 98 56 37 38 72 11 48
9	C	First note that f(0,n) = n + 1 f(2,1)= f(1,f(2,0))= f(1,f(1,1))= f(1,f(0,f(1,0)))= f(1,f(0,2))= f(1,3)= f(0,f(1,2))= f(0,f(0.f(1,1)))= f(0,f(0, f(0, f(1,0))))= f(0,f(0,f(0,f(0,1))))=5
10	A	Translate circuits to Boolean $\overline{\overline{a \wedge b} \wedge \overline{a \wedge b}} \leftrightarrow$ $(a \wedge b) \vee (a \wedge b) \leftrightarrow$ $a \wedge b$ Same as AND

		Solutions

11	A	int whosis (int &a, int& b, int c)
		{
		   a++;
		   b+=a;
		   c-=a++;
		   return a+b+c;
		}
		
		 int a=1, b=2, c=3;
		 cout << whosis(b,b,c);
		
		a is not used
		b=2 is renamed in the function with two names a and b.
		c=3 has same name in function.
		b=2
		a++ → b=3
		b+=a → b=6
		c-=a++ → c= -3 THEN a(b) changes to 7
		7 + 7 - 3 = 11

12	B	!A && B || C the order of operations is NOT, AND then OR so ((!A) && B) || C
13	E	A. allow the value of the variable *i* to be changed by the function. NO **const** does NOT permit changes B) conserve memory by returning unused memory to the system. NO Memory is conserved by using the &, but no memory is returned. C) allow for an increase in speed by reusing memory. NO again a function of the &
14	B	the operations * / and % all have the same precedence so the expression is evaluated like 13 - (((3 * 6) / 4) % 3) 18 / 4 = 4 4 % 3 = 1 13 -1 = 12

		Solutions
15	A	the line x = 3 does two things. First it sets x to 3. Then it returns the value 3 so that the if statement evalutes to true. output is 3

16	A	int x=30, y=40;
		if (x >= 0)
		{
		   if (x <= 100)
		   {
		      y = x*3;
		      ~~**if (y < 50)**~~
		         ~~**x /= 10;**~~
		   }
		   ~~else~~
		~~ y = x*2~~
		~~}~~
		~~else~~
		  ~~y = - x;~~
		since x >= 0 first if is done. The last else is not done ~~single~~.
		since x <=100 the second else is not done ~~double~~
		x*3 makes y=90 so the following if is skipped ~~**single bold**~~
		answer is x=30 y=90

17	C	I. ALL functions need headers NO II. functions must be declared BEFORE the are used. NO (even though most of a function can be listed after main, the prototype is the declaration and must come first III. TRUE III only
18	D	This program calculates the (N+1)th fibonacci number 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89

		Solutions

19	C	`  If (A<>B) Then`
		`    TrueOrFalse = A > B`
		`  Else`
		`    If (A>B) Then`
		`      TrueOrFalse = A > B`
		`    Else`
		`      TrueOrFalse = A < B`
		`    End If`
		`  End If`
		`End Function`
		`If A>B TRUE is returned by the first Then`
		`If A=B A<B (false) is returned by the second else`
		`If A < B (same as =)`
		`TrueOrFalse(6,5)`

20	D	A,B are examples not the most general case. A file is usually stored on some type of auxiliary storage. D. Variable
21	C	event
22	D	This is the "inside" loop of a bubble sort. There is a single traversal and at each step if the first element is bigger it is swapped with the next one
23	E	The first step changes List(0) to 1 (List(K) / List(0)) From that point everthing is divided by 1 so there are no other changes.

24	C	` 3141`
		`+2677`
		
		1+7 = 8 which is 10 in base 8
		put a **0** and carry 1
		(1) + 4 + 7 = 12 which is 14 in base 8
		put a **4** and carry a 1
		(1) + 1 + 6 = 8 (10 in base 8
		put a **0** and carry a 1
		(1) + 3 + 2 = 6
		6040

		Solutions

25	A	`Num = "4135"`
		`L = Len(Num)`
		`// L = 4`
		`S = 0`
		`T = 8`
		`For J = 1 to L`
		` X = Val(Mid$(Num, J, 1))`
		`//4  1  3  5`
		` S = S + X * T ^ (L - J)`
		`//S=0 + `**`4`**` * 8 ^ 3 = 2048`
		`//S=2048 + `**`1`**`*8^2= 2112`
		`//S=2112 + `**`3`**`*8^1= 2236`
		`//S=2236 + 5*8^0= 2241`
		`    Next J`

26	B	Since the list is in order, it is easy to find the 5 smallest elements. NO It is also easy to find the largest. It is in the last position. NO Insertion involves moving elements out of the way. Yes
27	C	Since the list is randomly ordered, the program must look at all *n* elements. $O(n)$
28	A	Visual Basic "fakes" Booleans by using integers. 0 is false, all other numbers are true. VB creates -1 when it evaluates a true statement x = 2 = 3 = 4 = 5 = 0 The first = is an assignment and it's order of operation is LAST. The other ='s are Boolean compares and happen left to right. 2=3 false (0) then 0=4 false (0) 0 = 5 false (0) THEN 0=0 true stored into an integer variable x x = -1 Print x results in -1
29	B	cout << works only with basic types and types for which it has been overloaded. III only
30	C	A1 gets 'B' A2 skips space and gets 3 n skips \n and gets 4