**Common mistakes**

1. This document shows all possible errors when tracing an algorithm in CS1010 questions.
3. **if** (-50 < i < -30)        //will still compile.-Wall will complain. it will process from left to right.)
4. is the same as          //it does the same when there is an assignment operator in the condition.
5. **if** ((-50 < i) < -30)
6. = **if** ((1 < -30)
7. = **if** (0)
9. ---------------------------------------------------------------------------------------------------------------------------
10. **for** ( ; i < 12 ; )       //will still compile, acts like a while loop instead
11. i++;
13. ---------------------------------------------------------------------------------------------------------------------------
14. **for** (i=0, i <5, i++);    //due to the ';' token at the end, will not process the next line.
15. printf(“%d”,i);     //will compile and print only the last value of the iteration of i.
17. ---------------------------------------------------------------------------------------------------------------------------
18. **double** a=1.5, b=1.55
19. **if** (a=b)                //the assignment will lead to the side effect of 'if (1.55)' which is TRUE.
20. printf(”a and b are equal”)
21. **else**
22. printf(”a and b are not equal”)
24. ---------------------------------------------------------------------------------------------------------------------------
25. **double** a=1.5, b=0
26. **if** (a=b)                //the assignment will lead to the side effect of 'if (0)' which is FALSE.
27. printf(”a and b are equal”)
28. **else**
29. printf(”a and b are not equal”)
31. ---------------------------------------------------------------------------------------------------------------------------
32. // To test the use of crappy operator styles
33. **void** operatorTest () {
34. **int** x;
35. printf("%d\n", x=10); // From this we can infer that x=10 just defaults and registers as 10
36. // OUTPUT: 10
37. printf("%d\n", x=0);
38. // OUTPUT: 0
39. printf("%d\n", x=10>11);
40. // OUTPUT: 0 because (x=10) gives 10 which is not more than 11 and thus returns 0 (FALSE)
41. printf("%d\n", y<(-1<3));
42. // OUTPUT: 1 whatever goes in the bracket will be processed first before left to right precedence is applied.
43. }
44. ---------------------------------------------------------------------------------------------------------------------------
45. i = (**int**)'A';
46. **switch** (i){
47. **case** 'A': printf("%d\n",i);     //will print all of the statements due to no "break;" statement
48. **case** 'B': printf("%d\n",i);     //fall through and prints this statement too
49. **case** 'C': printf("%d\n",i);     //fall through and prints this statement too
50. }
52. ---------------------------------------------------------------------------------------------------------------------------
53. **int** x=8,y=6,z=2;
54. x /= y %= z         //will compile. will give runtime error due to dividing zero.
56. ---------------------------------------------------------------------------------------------------------------------------
57. **for** (i=0; i < n; i = i\*2)    //will result in infinite loop due to "i = 0"
58. count++;
60. ---------------------------------------------------------------------------------------------------------------------------
61. **int** Continue = 5;   //is a valid initialisation statement. continue is a reserved term. not Continue.
63. ---------------------------------------------------------------------------------------------------------------------------
64. **int** 20cents\_digit;  //not a valid initialisation, first character cannot be a digit. '\_' is allowed as the first character.
66. ---------------------------------------------------------------------------------------------------------------------------
67. **int** x=3,y=5,z;
68. z = ++(x+y);        //will not compile. lvalue required as increment operand.
70. ---------------------------------------------------------------------------------------------------------------------------
71. sum -= --x - y      //there is an invisible parenthesis around ( x - y )
72. is the same as
73. x = x - 1; sum = sum - x + y;
74. ---------------------------------------------------------------------------------------------------------------------------
75. Weird multi operators:
77. a+-b is read as a + -b
78. a-+b. Is read a - +b
79. a---b is a-- - b
80. a+++b is a++ + b
82. ---------------------------------------------------------------------------------------------------------------------------
84. Initialising arrays
85. **int** arr[5] = {} == {0,0,0,0,0}
86. is the same as **int** arr[5] = {0}
88. **int** arr[5] //will reserve 5 memory spaces to put integers. will not allocate any values to them.
89. arr[][5]
90. need to define the number of columns **for** multidimensional arrays.
92. ---------------------------------------------------------------------------------------------------------------------------
93. STUFF to rectify
94. arrays
95. what happens when they print beyond the array limit - undefined
96. initialising arrays

99. **void** multiDimensionalArrayTest () {
100. /\*
101. case 1:
102. when we allocate more columns than we declare
104. case 2:
105. when we allocate only the row but not the column
107. case 3:
108. when we allocate only the column but called more rows than the ones we've declared
109. \*/
111. **int** arr[][4] = {{1, 2, 3}};
112. **int** i, k;
113. **for** (i = 0; i < 3; i++) {
114. **for** (k = 0; k < 4; k++) {
115. printf("%d\t", arr[i][k]);
116. }
117. printf("\n");
118. }
119. }
121. OUTPUT
122. 1 2 3 0
123. UNDEFINED UNDEFINED UNDEFINED UNDEFINED
124. UNDEFINED UNDEFINED UNDEFINED UNDEFINED
126. ---------------------------------------------------------------------------------------------------------------------------
127. **void** charTest () {
128. **char** c;
130. **while** (c != 'n') {
131. scanf(" %c", &c); // space added to keep new line from being registered as part of the char set.
132. printf("%c\n", c);
133. }
134. }
135. ---------------------------------------------------------------------------------------------------------------------------