計程二期中考

4/16/2004

1. 程式開發從寫作到執行會用到那些工具？請依使用順序列出，並任選四種工具寫出他們的作用。
2. 寫一單行程式用 rand( )隨機挑選並印出下列各組數字中之一個：
   a) 1, 2, 3, 4
   b) 6, 10, 14, 18, 22
   c) 1, 3, 7, 15, 31, 63 (本小題可用多行程式)
3. swap 是一個很有用的 function，呼叫 swap(a,b)會把 a, b 兩個 integer 變數內儲存的值互換。
   a) 請用 call by reference 的方式寫出 swap( )。
   b) 請用 call by value 加 pointer 的方式寫出 swap( )。
4. Local variable 中的 static variable 及非 static variable 有何不同。舉例說明。
5. 在一指向資料的 pointer 中，資料本身及 pointer 各自可為 constant 或 non-constant，因此共有四種可能，請舉例說明。
6. 請參考下列作業二之程式回答以下問題
   a) 如果 17 行改為 `if (rowsize == 5) direction = 0`，輸出有何不同？
   b) 修改程式使停止條件由出現連續兩個 1 改為連續兩個數字之和為 6。
   c) 修改程式使輸出由

```
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*
* *
* * *
* * * *
```

改為

```
        *
      * * *
    * * * * *
  * * * * * * *
* * * * * * * * *
  * * * * * * *
    * * * * *
      * * *
        *
      * * *
    * * * * *
  * * * * * * *
```

(Hint. 在換行後先印出正確數目的空格字元)

```
 1  int main()
 2  {
 3      int prev, cur = -1; // previous & current value
 4      int direction = 1;  // 1 means increasing, -1 means decreasing
 5      int rowsize = 1;    // number of outputs in a row
 6      int counter = 0;    // number of values already printed in a row
 7
 8      do { // loop until done
 9         prev = cur;     // keep track of previous value
10         cur = 1 + rand() % 6;   // generate current value
11         cout << setw( 10 ) << cur;
12         counter++;
13         if (counter == rowsize) {  // see if change of direction is needed
14            cout << endl;
15            counter = 0;
16            if (rowsize == 1) direction = 1;
17            if (rowsize == 5) direction = -1;
18            rowsize += direction;
19         }
```

```
20     } while ( prev != 1 || cur != 1);  // stop when both prev and cur are 1
21     return 0;  // indicates successful termination
22  } // end main
```

7. 請找出並修正下面的 binary search 之錯誤。

```
int binarySearch( const int b[], int searchKey, int low, int high)
{
   int middle;

   while ( low <= high ) {
      middle = low + high / 2;
      if ( searchKey = b[ middle ] )  // match
         return middle;
      else
         if ( searchKey < b[ middle ] )
            high = middle;  // search low end of array
         else
            low = middle;   // search high end of array
   }
   return -1;  // searchKey not found
} // end function binarySearch
```