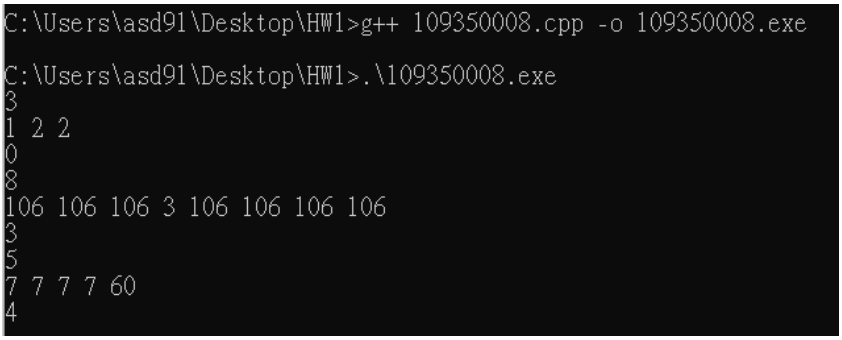
- Environment:

  OS: Windows

  Compiler: gcc/g++ version: 11.2.0

  IDE: Visual Studio (version: 16.11.5)

  可以用 cmd 開啟後執行，直接輸入 input



  也可以用 visual studio 開啟編譯後直接輸入 input

- Result:

  因為題目要求用 divided and conquer 且秤只能比較兩邊重量

  因此主要找假硬幣的方法為:

  1. 把陣列分成 2 半，比較兩邊的重量
  2. 接著跑可能出現假硬幣的那一邊，而另一邊隨意找一個當真實重量 (cmp) 以用於之後比較假硬幣
  3. 若是當下跑的個數為奇數個，則先去掉中間(mid)，比較 start 到 mid-1 以及 mid+1 到 end 的重量。若是一樣，則代表 mid 為假硬幣，回傳其位置，若不一樣，則繼續跑 2. 以及 1.
  4. 直到分成剩下 3 個或 2 個時
     - 若為 3 個: 直接比較 3 個看哪一個重量和其他 2 個不一樣，回傳不一樣重量的 index。
     - 若為 2 個: 比較看哪個和 cmp 不一樣，再回傳其位置。

  因為不知道假硬幣是比較重還是比較輕，因此分為假設假硬幣是較輕或是

較重的情況。 先跑假設假硬幣是比較重的情況，若假硬幣的重量真的比較重，就可以直接跑上面的方法直到找出假硬幣位置。 但若是假硬幣是較輕的，則在第二次分開比較時，因為假硬幣在另外一邊，而較重這邊會有 2 種情況:

- 個數為偶數時:

  第二次分開比較兩邊重量會一樣，因此可知假硬幣為較輕，從頭再跑較輕的假設。

- 個數為奇數時:

  第二次分開跑方法 3. 時，去掉中間後兩邊的重量一定會一樣，但不可回傳 mid，因此拿隨便一個不是 mid 的做為比較，若一樣，則可知假硬幣較較輕，從頭再跑較輕的假設。

而跑較輕假設時因為已經跑過較重假設，所以可知假硬幣重量一定較輕，因此可直接照上面的方法直到找出假硬幣位置。

Time complexity:

Time complexity of whole algorithm:
主要取決於分兩邊跑遞迴，以及在比較 2 邊時要跑迴圈算重量
$T(n) = 2T(n/2) + n \Rightarrow T(n) = \Theta(nlgn)$ by case2 of master method

time complexity of using scale:
$T(n) = 2T(n/2) + C$ (C is constant) $\Rightarrow T(n) = \Theta(n)$ by case1 of master method