實作目標：比較sorted array、array of sorted arrays、skip list的range query所需時間。

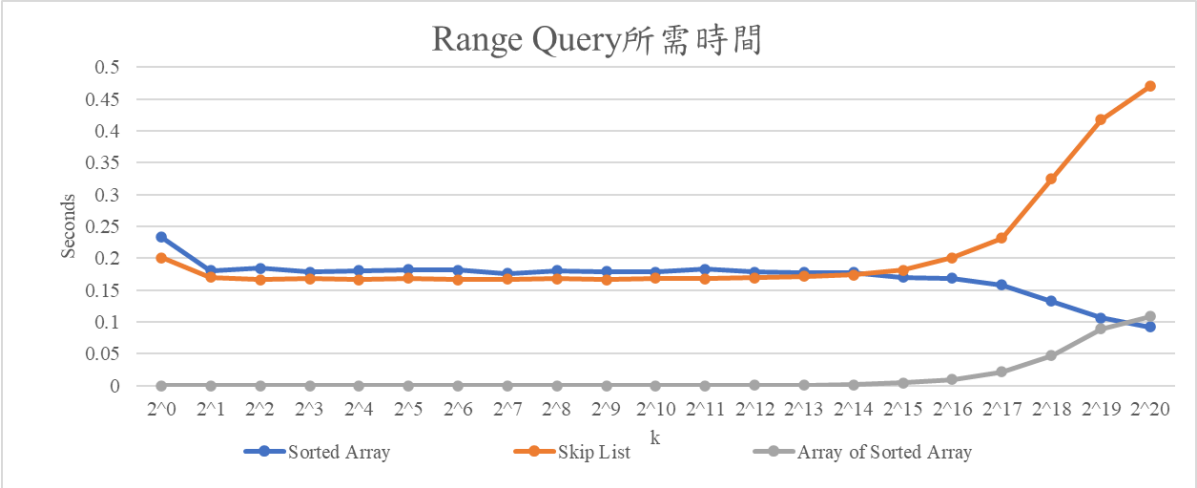
實作以下3個function:

1. range_query:
   a. input: 兩個正整數a和b。
   b. output: 資料結構中所有介於a和b（大於等於a且小於等於b）的資料。這些資料須存放在一個multiset中（multiset的實作不限，如array或linked list）。
2. equivalent:
   a. input: 兩個multisets S_1與S_2。
   b. output: yes if S_1和S_2有相同元素，而且每個元素在S_1出現的次數等於該元素在S_2出現的次數；
      no otherwise。
3. print:
   a. input: 一個multisets S。
   b. 在stdout輸出S的內容。

實驗步驟：

1. $n = 2^{20} - 1, k = 2^0, 2^1, 2^2, \ldots, 2^{20}$ (可視電腦執行速度調整n與k的值)
2. 產生n個數，每個數字都從$0 \sim n-1$隨機產生（每個數字被選到的機率都是$\frac{1}{n}$）。
3. 將上述n個數存在檔案中，每行存一個數字。
4. 讀檔並將n個數存入三個資料結構中。
5. 隨機從$0 \sim n-1$產生a（每個數字被選到的機率都是$\frac{1}{n}$）。
6. 在三個資料結構執行range_query$(a, a+k)$，並測量時間。
7. 利用equivalent檢查range_query結果是否一致。

一、折線圖與解釋實驗結果



- 總結：Array of Sorted Array Range Query 的執行速度最快，k≦2^14，Skip List 的速度較快，search 範圍更大後，Sorted Array 反而比較快。
- Skip List vs Sorted Array 解釋：Skip List 通過多層的連結允許跳過不必要的節點，實現 Search time complexity O(log n)，因此比 binary search Sorted Array 快，但 search 範圍更大後，binary search 加速效果可能比 Skip List 的多層結構有效，使 binary search Sorted Array 較快。
- Array of Sorted Array 解釋：本來每個 array 已經排序好，加上 binary search 使得 range query search 最快。