ชื่อ ..... รหัสนักศึกษา.....

05506003 Programming Fundamentals

Lab Week 8

วัตถุประสงค์

A. นักศึกษาสามารถประยุกต์ใช้การเขียนโค้ดเพื่อตัดสินใจและแบบวนลูปได้อย่างแม่นยำ

B. นักศึกษาสามารถเขียน method และเรียกใช้ได้

B. นักศึกษาใช้ทักษะในการเขียนโปรแกรมเพื่อแก้โจทย์ที่ซับซ้อน

In class

กิจกรรมที่ 1 เขียน static int secondLargest(int ... arr) { ... }

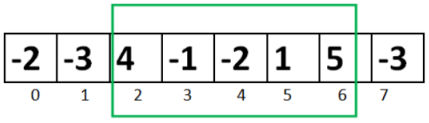
กิจกรรมที่ 2 เขียน ProFun08_Q2_xxyyyy.java

Largest Sum Contiguous SubArray Problem คือ ค่าที่มากที่สุดของผลรวมของ subset ของ array

การใช้กลยุทธ์ brute force คือ หาผลรวมของ ค่า start stop ทุกค่าที่เป็นไปได้ดังนี้

```
                public static void main(String[] args) {
                  int [] data = {-2,-3,4,-1,-2,1,5,-3};
                  q2_1_BF(data);
                }
/* 10 */        static void q2_1_BF(int ... data) {
/* 20 */          int max = Integer.MIN_VALUE;
/* 30 */          int sum /*, slow_sum */ ;
/* 40 */          int start, stop;
/* 50 */          start = stop = 0;
/* 60 */          int numCases = 0;
/* 70 */          //all pair of i,j
/* 80 */          for (int i = 0; i < data.length - 1; i++) {
/* 90 */              sum = 0;
/*100 */              for (int j = i; j < data.length; j++) {
/*110 */                  /* your code */
/*120 */                  // slow_sum = 0;
/*130 */                  // for (int k = i; k <= j; k++)
/*140 */                  //     slow_sum += data[k];
/*150 */                  printf("case %d for start,stop = %d,%d ->
                                    sum = %d",++numCases,i,j,sum);
/*160 */                  if (sum > max) {
/*170 */                      start = i;
/*180 */                      stop = j;
/*190 */                      max = sum;
/*200 */                     println(" ** new max **");
/*210 */                  } else {
/*220 */                     println();
/*230 */                  }
/*240 */              }
```

Largest Subarray Sum Problem



4 + (-1) + (-2) + 1 + 5 = 7

Maximum Contiguous Array Sum is 7

https://www.geeksforgeeks.org/largest-sum-contiguous-subarray/

ทุก i,j คือ subset โดย i คือดัชนีของช่องที่เริ่ม และ j คือดัชนีของช่องที่สิ้นสุด ดังนั้นเราสามารถใช้ for k เพื่อหาผลรวมของแต่ละ i,j

แต่เราสามารถตัดเวลาสำหรับ for k ทิ้งได้ เพราะทุกรอบของ j คือ stop ที่ยาวขึ้นจึงสามารถสะสมเป็น sum ของ j ถัดไปจนกว่าจะเริ่ม i ค่าถัดไป

Q2_1 เติม q2_1_BF() ให้สมบูรณ์

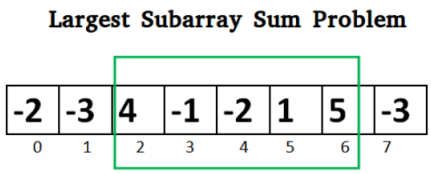
ชื่อ ..... รหัสนักศึกษา.....

หมายเหตุ Kadane Algorithm สามารถหาค่าด้วย O(n) โดยข้าม subarray ที่มีค่า sum เป็นลบ และ forward index ทางซ้ายข้าม subarray ที่เพิ่งประมวลผลมา

Kadane's Algorithm สำหรับแก้ปัญหานี้ ระบุไว้ดังนี้

```
// Initialize:
//     max_so_far = INT_MIN
//     max_ending_here = 0

// Loop for each element of the array
//   (a) max_ending_here = max_ending_here + a[i]
//   (b) if(max_so_far < max_ending_here)
//             max_so_far = max_ending_here
//   (c) if(max_ending_here < 0)
//             max_ending_here = 0
// return max_so_far
```



https://www.geeksforgeeks.org/largest-sum-contiguous-subarray/

ให้ นศ. ศึกษาว่า (index) start และ stop นั้นถูกต้องหรือไม่

```
48  static void q2_2_kadane(int ... data) {
49      int start, stop;
50      start = stop = 0;
51      int max_so_far = Integer.MIN_VALUE;
52      int max_ends_here = 0;
53      // int len = 0;
54      int mark_i_as_start = 0;
55
56      for (int i = 0; i < data.length; i++) {
57          max_ends_here += data[i];
58          //len++;
59          if (max_so_far < max_ends_here) {
60              start = mark_i_as_start;    // reset via if (max_ends_here < 0)
61              stop = i;
62              max_so_far = max_ends_here;
63              System.out.printf(format:"%d %d is new max = %d%n",start,stop,max_so_far);
64          }
65          if (max_ends_here < 0) {
66              max_ends_here = 0;  // negative cell cannot be the start cell
67              //len = 0;
68              mark_i_as_start = i + 1;      // fast forward mark_i_as_start to skip all subset of ending at i
69          }
70      }
71      System.out.printf(format:"%d %d = %d%n",start,stop,max_so_far);
72  }
```

Homework

กิจกรรมที่ 3

เขียน static boolean q3_common_element(int [] a, int [] b) { ... } โดย a และ b ต่างเป็นอาร์เรย์ที่ข้อมูลเรียงจากน้อยไปมาก (เช่น a = {2,3,5,7} b = {4,6,7,8}) รีเทิร์น true หากมีค่าที่ปรากฏทั้งใน a และ b

ชื่อ ..... รหัสนักศึกษา.....

กิจกรรมที่ 4

เขียน static void topK(int ... data) { } โดยมี intArr เป็นอาร์เรย์ 1 มิติ ขนาด 10 ช่อง มีค่าเริ่มต้นของทุกช่องเท่ากับ 0 จากนั้น ให้รับค่ามาจากคีย์บอร์ดทีละ 1 ค่า เป็นจำนวนเต็ม เพื่อใส่ลงในอาร์เรย์ intArr (ปรับเป็นรับจาก int [ ] data) ดังนี้ (ตัวอย่างใส่ 0 ปิดท้ายให้ดังนั้น สามารถตรวจได้ว่าหากพบศูนย์ให้จบการประมวลผล)

1. หากในอาร์เรย์ไม่มีตัวเลขอื่นนอกจาก 0 อยู่เลย และเลขที่รับมาเป็นจำนวนเต็มบวก ให้ใส่ค่าที่ตำแหน่งแรก

2. หากค่าที่รับมาเป็นจำนวนเต็มบวก และค่าในอาร์เรย์มีตัวเลขอื่นนอกจาก 0 ให้นำไปแทรกที่หลังตัวเลขที่น้อยที่สุด ที่มากกว่าค่าที่รับมา หากไม่มี ให้แทรกที่ตำแหน่งแรก

3 หากมีการแทรกค่า ให้เลื่อนข้อมูลในอาร์เรย์ทั้งหมด นับตั้งแต่ตำแหน่งที่ถูกแทรกไปทางขวา 1 ช่อง หากเลขที่ขยับออกไปนอกขอบเขตของอาร์เรย์ให้ถือว่าเลขนั้นหายไป

4. ให้รับค่าเรื่อย ๆ จนกว่าจะมีการใส่ตัวเลขจำนวนเต็มลบ หรือศูนย์ (ข้อมูลที่ป้อนเข้ามา จะมีกี่ตัวก็ได้ จนกว่าจะป้อนข้อมูลจำนวนเต็มลบหรือศูนย์)

5. ให้แสดงอาร์เรย์ปัจจุบัน ใน caller เมื่อมีการแทรกเลขสำเร็จทุกครั้ง และ สุดท้าย ดังตัวอย่าง

insert 5 -> [5, 0, 0, 0, 0, 0, 0, 0, 0, 0]
insert 3 -> [5, 3, 0, 0, 0, 0, 0, 0, 0, 0]
insert 8 -> [8, 5, 3, 0, 0, 0, 0, 0, 0, 0]
insert 4 -> [8, 5, 4, 3, 0, 0, 0, 0, 0, 0]
insert 10 -> [10, 8, 5, 4, 3, 0, 0, 0, 0, 0]
insert 3 -> [10, 8, 5, 4, 3, 3, 0, 0, 0, 0]
insert 1 -> [10, 8, 5, 4, 3, 3, 1, 0, 0, 0]
insert 5 -> [10, 8, 5, 5, 4, 3, 3, 1, 0, 0]
insert 9 -> [10, 9, 8, 5, 5, 4, 3, 3, 1, 0
insert 7 -> [10, 9, 8, 7, 5, 5, 4, 3, 3, 1]
insert 2 -> [10, 9, 8, 7, 5, 5, 4, 3, 3, 2]
final [10, 9, 8, 7, 5, 5, 4, 3, 3, 2]

สำหรับ PEARLS ให้เพิ่ม จำนวน input ไว้เป็นค่าแรก แทนการจบด้วยค่า <= 0 เช่น 11 5 3 8 4 10 3 1 5 9 7 2

คำสั่ง เขียน ProFun08_ xxyyyy.java (q3 และ q4)