Lab Worksheet

ชื่อ-นามสกุล นาย ภคพล อยู่ยืน **รหัสนศ.** 663380226-7 **Section. 1**

Lab#7 – White-box testing

วัตถุประสงค์การเรียนรู้

1. ผู้เรียนสามารถออกแบบการทดสอบแบบ White-box testing ได้
2. ผู้เรียนสามารถวิเคราะห์ปัญหาด้วย Control flow graph ได้
3. ผู้เรียนสามารถออกแบบกรณีทดสอบโดยคำนึงถึง Line coverage ได้
4. ผู้เรียนสามารถออกแบบกรณีทดสอบโดยคำนึงถึง Block coverage ได้
5. ผู้เรียนสามารถออกแบบกรณีทดสอบโดยคำนึงถึง Branch coverage ได้
6. ผู้เรียนสามารถออกแบบกรณีทดสอบโดยคำนึงถึง Condition coverage ได้
7. ผู้เรียนสามารถออกแบบกรณีทดสอบโดยคำนึงถึง Branch and Condition coverage ได้

โจทย์: CLUMP COUNTS

Clump counts (https://codingbat.com/prob/p193817) เป็นโปรแกรมที่ใช้ในการนับการเกาะกลุ่มกันของข้อมูลภายใน Array โดยการเกาะกลุ่มกันจะนับสมาชิกใน Array ที่อยู่ติดกันและมีค่าเดียวกันตั้งแต่สองตัวขึ้นไปเป็นหนึ่งกลุ่ม เช่น

[1, 2, 2, 3, 4, 4] ⟶ 2

[1, 1, 2, 1, 1] ⟶ 2

[1, 1, 1, 1, 1] ⟶ 1

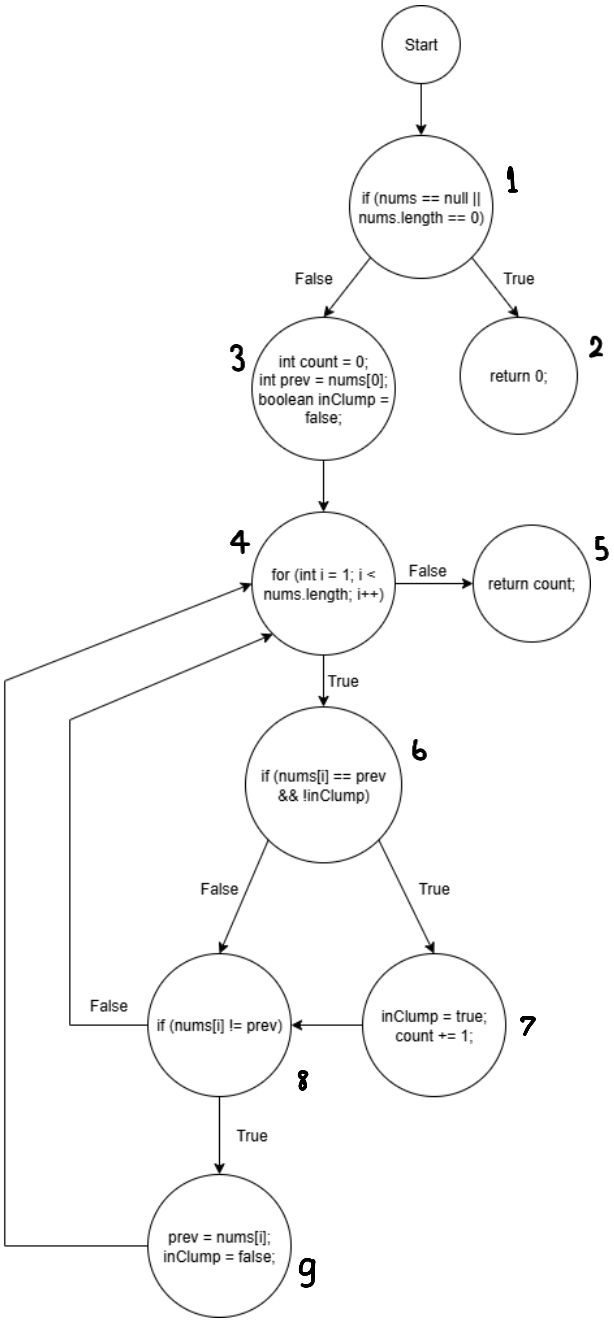
ซอร์สโค้ดที่เขียนขึ้นเพื่อนับจำนวนกลุ่มของข้อมูลที่เกาะอยู่ด้วยกันอยู่ที่ https://github.com/ChitsuthaCSKKU/SQA/tree/2025/Assignment/Lab7 โดยที่ nums เป็น Array ที่ใช้ในการสนับสนุนการนับกลุ่มของข้อมูล (Clump) ทำให้ nums เป็น Array ที่จะต้องไม่มีค่าเป็น Null และมีความยาวมากกว่า 0 เสมอ หาก nums ไม่เป็นไปตามเงื่อนไขที่กำหนดนี้ โปรแกรมจะ return ค่า 0 แทนการ return จำนวนกลุ่มของข้อมูล

แบบฝึกปฏิบัติที่ 7.1 CONTROL FLOW GRAPH

จากโจทย์และ Source code ที่กำหนดให้ (CountWordClumps.java) ให้เขียน Control Flow Graph (CFG) ของเมธอด countClumps() จากนั้นให้ระบุ Branch และ Condition ทั้งหมดที่พบใน CFG ให้ครบถ้วน

ตอบ

Lab instruction



Branch:

1_True = return 0;

1_False = int count = 0;

int prev = nums[0];

boolean inClump = false;

4_False = return count;

4_True = if(nums[i] == prev && !inClump)

6_True = inClump = true;

count += 1;

6_False = if(nums[i] != prev)

8_True = prev = nums[i];

inClump = false;

8_False = i++; //for(int i=1; i<nums.length; i++)

Condition:

A = nums = null

B = nums.length == 0

C = int i < nums.length

D = nums[i] == prev

E = **!**inClump

F = nums[i] != prev

แบบฝึกปฏิบัติที่ 7.2 LINE COVERAGE

1. จาก Control Flow Graph (CFG) ของเมธอด countClumps() ในข้อที่ 1 ให้ออกแบบกรณีทดสอบเพื่อให้ได้ Line coverage = 100%
2. เขียนกรณีทดสอบที่ได้ พร้อมระบุบรรทัดที่ถูกตรวจสอบทั้งหมด
3. แสดงวิธีการคำนวณค่า Line coverage

ตอบ

Test Case No.	Input(s)	Expected Result(s)	Path and Branch
1	null	0	Line No.: 6,7
2	[ ]	0	Line No.: 6,7
3	[1,1,1]	1	Line No.: 6,10,11,12,14,15,16,17,20,25
4	[0,0,1,2,2]	2	Line No.: 6,10,11,12,14,15,16,17,20,21,22,25

Line coverage = $\left(\frac{13}{13}\right) \times 100$ = **100%**

แบบฝึกปฏิบัติที่ 7.3 BLOCK COVERAGE

1. จาก Control Flow Graph (CFG) ของเมธอด countClumps() ในข้อที่ 1 ให้ออกแบบกรณีทดสอบเพื่อให้ได้ Block coverage = 100%
2. เขียนกรณีทดสอบที่ได้ พร้อมระบุ Block ที่ถูกตรวจสอบทั้งหมด
3. แสดงวิธีการคำนวณค่า Block coverage

ตอบ

Test Case No.	Input(s)	Expected Result(s)	Path and Branch
5	[ ]	0	Block: 1,2
6	[1,1]	1	Block: 1,3,4,6,7,8,5
7	[2,3]	0	Block: 1,3,4,6,8,9,5
8	[4,4,5,6,6]	2	Block: 1,3,4,6,7,8,9,5

Block coverage = $\left(\frac{9}{9}\right) \times 100$ = **100%**

แบบฝึกปฏิบัติที่ 7.4 BRANCH COVERAGE

4. จาก Control Flow Graph (CFG) ของเมธอด countClumps() ในข้อที่ 1 ให้ออกแบบกรณีทดสอบเพื่อให้ได้ Branch coverage = 100%
5. เขียนกรณีทดสอบที่ได้ พร้อมระบุ Path และ Branch ที่ถูกตรวจสอบทั้งหมด
6. แสดงวิธีการคำนวณค่า Branch coverage

ตอบ

*x_T = Branch ที่ x = True , x_F = Branch ที่ x = False

Test Case No.	Input(s)	Expected Result(s)	Path and Branch
9	[ ]	0	Path: **1-2** Branch: **1_T**
10	[1]	0	Path: **1-3-4-5** Branch: **1_F**, **4_F**
11	[1,1]	1	Path: **1-3-4-6-7-8-4-5** Branch: **1_F, 4_T, 6_T**, **8_F**, **4_F**
12	[2,3]	0	Path: **1-3-4-6-8-9-4-5** Branch: **1_F**, **4_T**, **6_F**, **8_T**, **4_F**
13	[0,0,1]	1	Path: **1-3-4-6-7-8-4-6-8-9-4-5** Branch: **1_F**, **4_T**, **6_T**, **6_F**, **8_F**, **8_T**, **4_F**

Branch coverage = $\left(\frac{8}{8}\right) \times 100$ = **100%**

แบบฝึกปฏิบัติที่ 7.5 CONDITION COVERAGE

1. จาก Control Flow Graph (CFG) ของเมธอด countClumps() ในข้อที่ 1 ให้ออกแบบกรณีทดสอบเพื่อให้ได้ Condition coverage = 100%
2. เขียนกรณีทดสอบที่ได้ พร้อมระบุ Path และ Condition ที่ถูกตรวจสอบทั้งหมด เช่น Condition A = T และ Condition B = F
3. แสดงวิธีการคำนวณค่า Condition coverage

ตอบ

Condition:

A = nums = null , **B** = nums.length == 0 , **C** = int i < nums.length , **D** = nums[i] == prev , **E** = **!**inClump , **F** = nums[i] != prev

Test Case No.	Input(s)	Expected Result(s)	Path and Condition
14	null	0	Path: 1-2 Condition: **A**
15	[ ]	0	Path: **1-2** Condition: **B**
16	[1,1]	1	Path: **1-3-4-6-7-8-4-5** Condition: **C**, **D**, **E**
17	[2,3]	0	Path: **1-3-4-6-8-9-4-5** Condition: **C, E, F**

Condition coverage = $\left(\frac{6}{6}\right) \times 100$ **= 100%**

แบบฝึกปฏิบัติที่ 7.6 BRANCH AND CONDITION COVERAGE (C/DC COVERAGE)

1. จาก Control Flow Graph (CFG) ของเมธอด countClumps() ในข้อที่ 1 ให้ออกแบบกรณีทดสอบให้ได้ C/DC coverage = 100%
2. เขียนกรณีทดสอบที่ได้ พร้อมระบุ Path, Branch, และ Condition ที่ถูกตรวจสอบทั้งหมด
3. แสดงวิธีการคำนวณค่า C/DC coverage
4. เขียนโค้ดสำหรับทดสอบตามกรณีทดสอบที่ออกแบบไว้ด้วย JUnit และบันทึกผลการทดสอบ

ตอบ

Condition:

A = nums = null , B = nums.length == 0 , C = int i < nums.length , D = nums[i] == prev , E = !inClump , F = nums[i] != prev

Test Case No.	Input(s)	Expected Result(s)	Actual Result(s)	Path, Branch, and Condition
18	null	0	Pass/Fail: Pass	Path: **1-2** Branch: **1_T** Condition: **A**
19	[ ]	0	Pass/Fail: Pass	Path: **1-2** Branch: **1_T** Condition: **B**
20	[1]	0	Pass/Fail: Pass	Path: **1-3-4-5** Branch: **1_F, 4_F** Condition: **-**
21	[1,1]	1	Pass/Fail: Pass	Path: **1-3-4-6-7-8-4-5** Branch**: 1_F, 4_T, 6_T, 4_F, 8_F** Condition: **C, D, E**

22	[2,3]	0	Pass/Fail: Pass	Path: **1-3-4-6-8-9-4-5** Branch: **1_F**, **4_T**, **6_F**, **8_T**, **4_F** Condition: **C**, **E**, **F**
23	[3,3,4]	1	Pass/Fail: Pass	Path: **1-3-4-6-7-8-4-6-8-9-4-5** Branch: **1_F, 4_T, 6_T, 8_F, 6_F, 8_T, 4_F** Condition: **C, D, E, F**

C/DC coverage = $\left(\frac{14}{14}\right) \times 100$ = **100%**