บทที่ 4

การทดลองและผลการทดลองของกรณีทดสอบ

ในบทนี้จะกล่าวถึงการนำวิธีวิเคราะห์ที่ได้ศึกษามาต่างๆ มาทำการเขียนโปรแกรมสำหรับทำการตรวจสอบในวิธีการของแบบ White-box Testing และ Black-box Testing เพื่อตรวจหาเส้นทางการทำงานของโปรแกรมที่นำมาทำการทดสอบ และตรวจสอบถึงผลลัพธ์ที่ได้ของโปรแกรมว่ามีการทำงานที่ถูกต้องหรือไม่

4.1 การทดสอบโปรแกรมโดยวิธี White Box Testing

ในการทดสอบโปรแกรมนั้น Software ที่ทำหน้าที่ทดสอบนั้นจะทำหน้าที่ทดสอบแบบ white-box testing กล่าวคือ จะทำการทดสอบโปรแกรมโดยจะทำการทดสอบในตัวโครงสร้างของโปรแกรม ดูลำดับการทำงานของโปรแกรม และมองดู path ของโปรแกรมที่นำมาทดสอบว่าเป็นอย่างไร มีกี่เส้นทาง รวมถึงมองดูค่าต่างๆที่ใช้เป็นตัวเลือกของโปรแกรมใน path ต่างๆ นอกจากนี้โปรแกรมยังจะทำการคำนวณค่าต่างๆมาป้อนให้กับโปรแกรมเพื่อหาว่า โปรแกรมนี้สามารถเดินทางผ่านทุกๆ path ที่สร้างขึ้นมาได้หรือไม่ และหาเส้นทางที่ไม่สามารถผ่านได้ หรือเส้นทางที่ทำให้เกิด error

โปรแกรมทดสอบ Software นี้ จะถูกสร้างขึ้นจากภาษา Java ซึ่งมีคุณสมบัติในการจัดการกับ input และ output stream ในรูปแบบต่างๆได้ดี และยังสามารถสร้าง process เพื่อสั่งการให้โปรแกรมใดๆทำงานในระหว่างที่โปรแกรมตรวจสอบทำงานอยู่ได้ ทำให้สามารถสั่งให้ตรวจสอบโปรแกรมได้ทั้งทางโครงสร้างและผลลัพธ์ที่เกิดจากการทำงานของโปรแกรมนั้นๆได้

ในการทดลองนั้น จะต้องพัฒนาโปรแกรมให้สามารถติดต่อกับไฟล์ภายนอกได้ กล่าวคือ โปรแกรมต้องสามารถรับ Input ที่เป็นไฟล์สตรีมได้ ซึ่งไฟล์ที่รับมานั้น ก็คือ source code ที่จะนำมาทดสอบ ซึ่งตัวโปรแกรมจะสร้าง buffer เพื่อเก็บข้อมูลหรือ source code ที่เข้ามาจาก input stream ทั้งหมดก่อนที่จะนำไปวิเคราะห์ค่าต่อไป

ในขอบเขตของการทดสอบนั้น ระบุไว้ว่า โปรแกรมที่จะทดสอบต้องเป็นโปรแกรมในภาษา C,C++ ซึ่งในการสร้าง path ของโปรแกรมก็จะต้องอ้างอิงจากหลักการเขียน และ syntax ของภาษา

นี้ ซึ่งในเบื้องต้นได้กำหนดขอบเขตไว้ว่า ให้มีเงื่อนไข และ loop ต่างๆได้เพียงแค่ if, else, for ,while ซึ่งโปรแกรมจะต้องเงื่อนไขค่าเหล่านี้มาพิจารณาในการสร้าง path ต่อไป

ในการสร้าง Path นั้น เราจะสร้างออกมาเป็น graph ซึ่งมี node แต่ละ node แทนความหมายของ statement, start-loop และ end-loop ซึ่งในแต่ละ node จะมีการเก็บค่าต่างๆไว้คือ

1. ใน statement – จะเก็บ code ในบรรทัดนั้นๆ เป็น string, เก็บค่าเลขบรรทัดเพื่อบอกว่า statement นั้นอยู่ตรงไหนเป็น integer และค่าของ node ต่อไปเป็นค่า integer ที่เป็นดรรชนีชี้node ต่างๆใน array ของ node

2. ใน start-loop – จะเก็บค่าของเงื่อนไขต่างเป็นเช่น if เป็นต้น string และเก็บสมการของเงื่อนไขเพื่อใช้ในการพิจารณา input ในวิธี black-box testing เป็น string และเก็บค่าของ node ต่อไปเช่นเดียวกับ statement

3. ใน end-loop – จะเป็นตัวกำหนดว่า loop หรือเงื่อนไขใดๆจบลง ใช้เพื่อจำกัดขอบเขตของ loop ต่าง จะเก็บค่าของ node ต่อไป

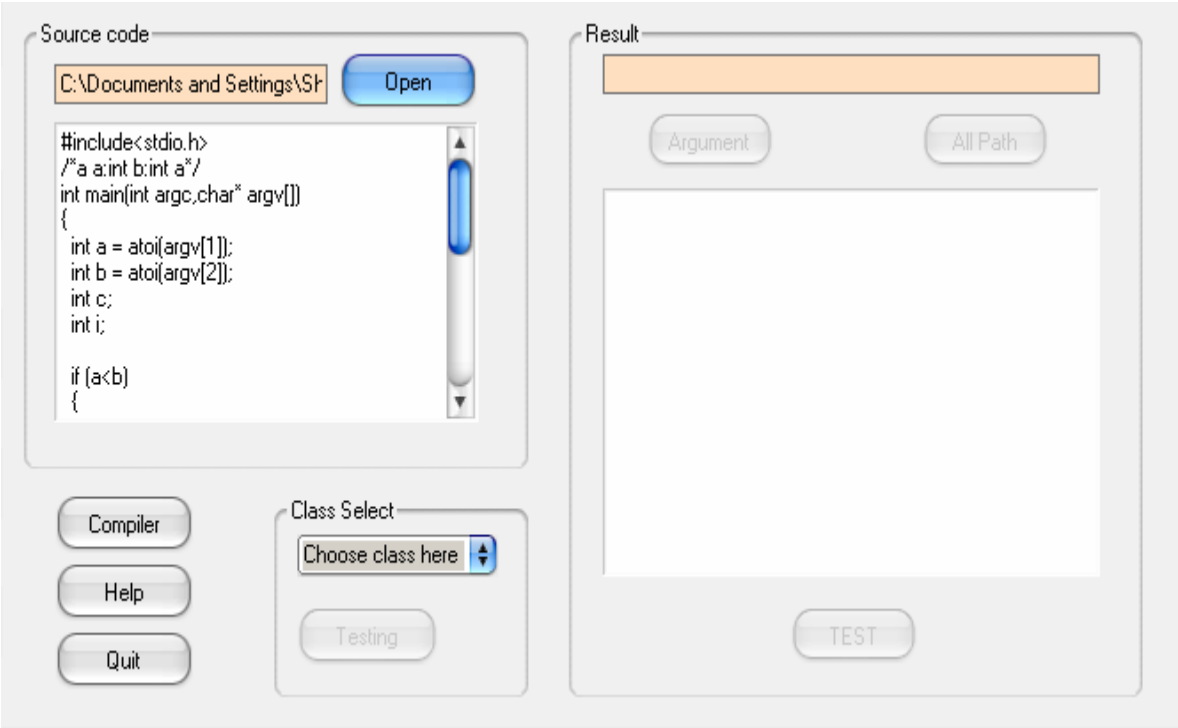
4.2 การทดลองและผลการทดลองในส่วนของวิธี White-box Testing

ในการทดลองในส่วนของ White-box Testing นั้น ก็จะทำการเปิดไฟล์ที่จะทำการทดสอบ โดยไฟล์ที่ใช้ทำการทดสอบในการทำวิจัยนี้ชื่อว่า Test.C

Source Code ไฟล์ Test.C ที่จะใช้มาทำการทดสอบ

```
#include<stdio.h>
/*a a:int b:int a*/
int main(int argc,char* argv[])
{
  int a = atoi(argv[1]);
  int b = atoi(argv[2]);
  int c;
  int i;

  if (a<b)
  {
printf("cid:0\n");

    c = b;
  }
  while (0)
  {
printf("cid:1\n");

    c = a;
  }

  printf("%d",c);
  return c;
}_
```

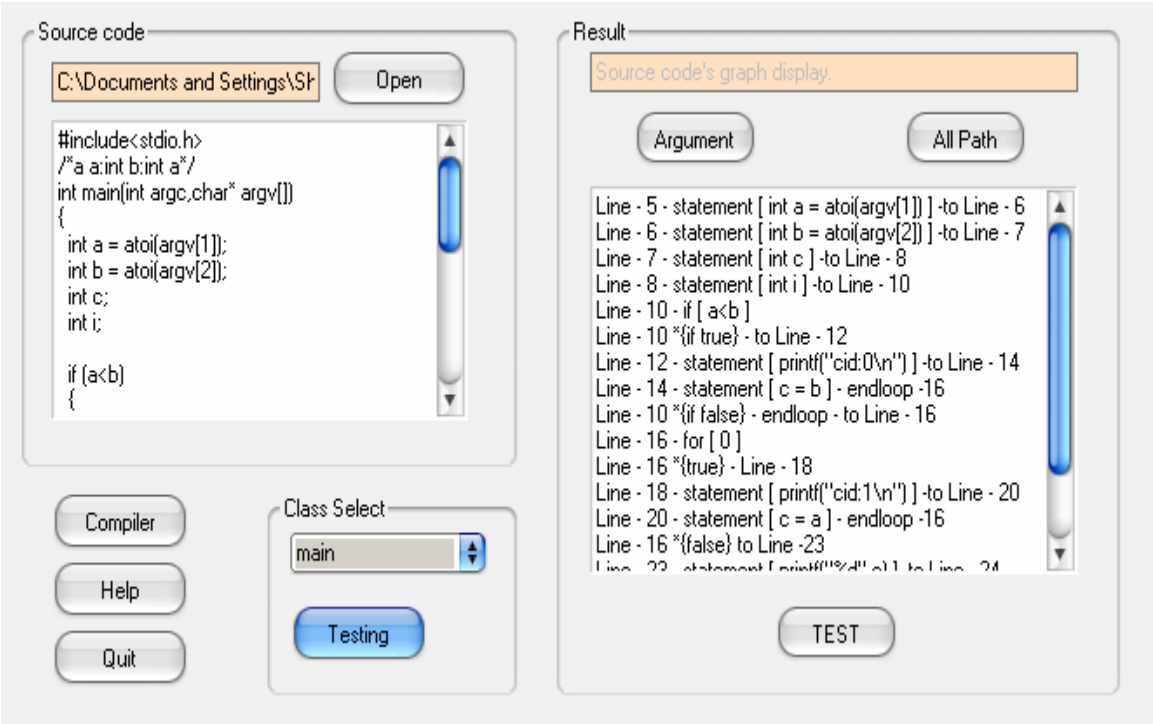
โปรแกรมก็จะแสดงให้เห็น Source Code ของโปรแกรม Test.C ให้เห็นขึ้นมาดังรูปที่ 4.1



รูปที่ 4.1 รูปแสดงผล Source Code ที่ได้จากการเปิดไฟล์ที่จะทำการทดสอบ

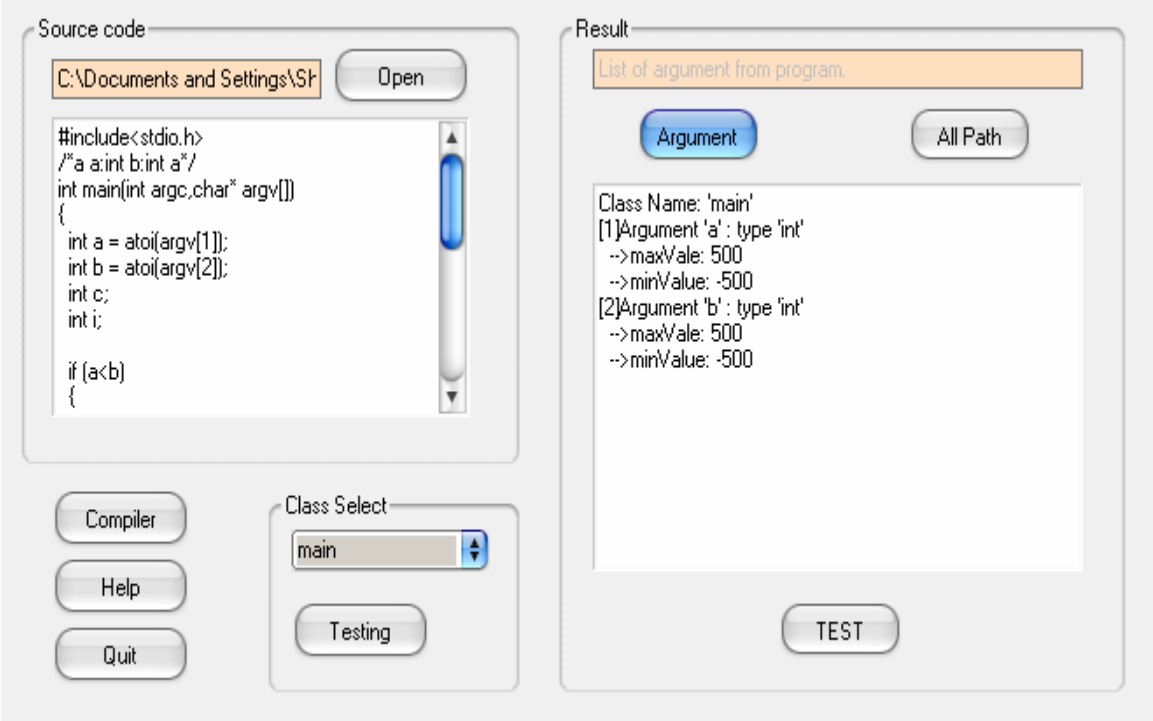
จากนั้นก็ให้ทำการเลือก Class ที่จะนำไปทำการทดสอบ โดยในการทดสอบนี้จะทำการทดสอบใน Class main

หลังจากเลือก Class ที่ทำการทดสอบ จากนั้นก็จะแสดงผลการทดสอบที่ได้ โดยผลการทดสอบที่แสดงออกมา จะเป็นการแสดงให้เห็นถึง Control Flow Graph ที่อยู่ในรูปของ Text Mode โดยจะอธิบายว่าใน Source Code ที่นำมาทดสอบมีเส้นทางการทำงานอย่างไรบ้าง ดังแสดงให้เห็นในรูปที่ 4.2



รูปที่ 4.2 รูปแสดงผล Control Flow Graph ที่ได้จากการทดสอบในรูปของ Text Mode

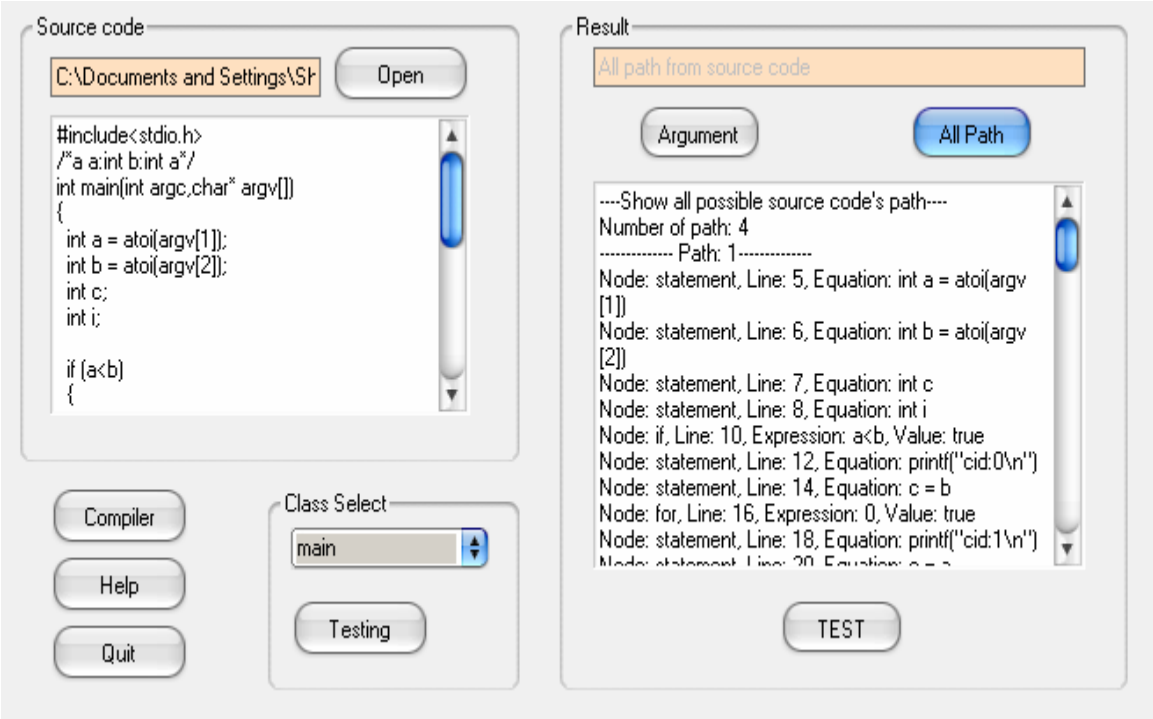
และสามารถแสดงถึง Argument ที่ใช้ในการทดสอบว่าเป็น Argument ชนิดใดและมีค่าขอบเขตมากสุดและน้อยสุดเป็นเท่าไร ดังแสดงให้เห็นในรูปที่ 4.3



รูปที่ 4.3 รูปแสดงผล Argument ที่ได้จากการทดสอบ

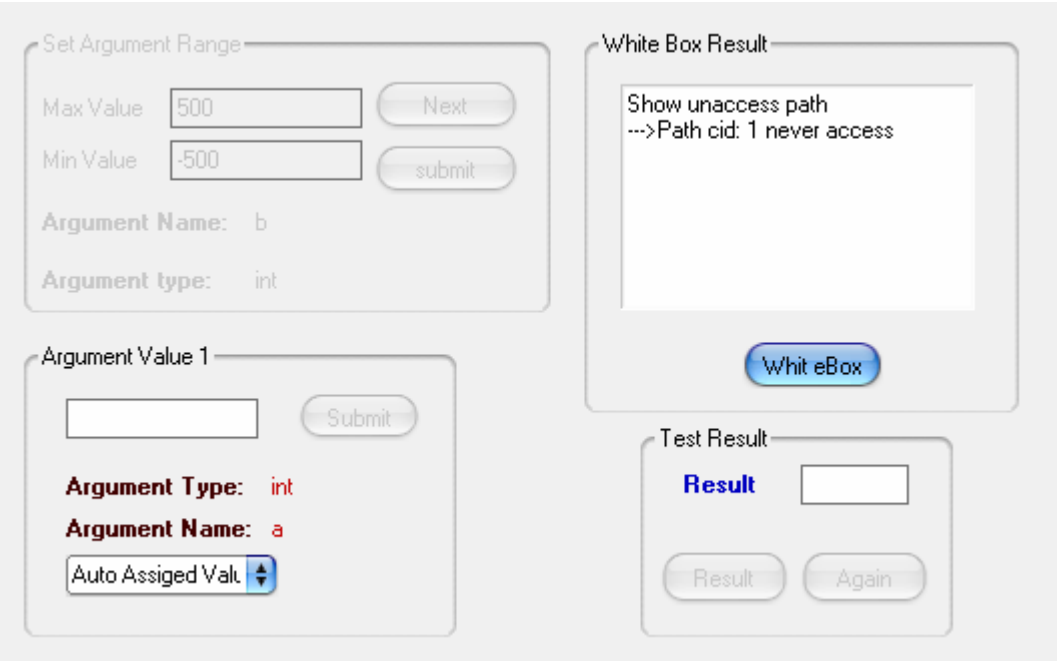
และแสดงให้เห็นถึงเส้นทางการทำงานของโปรแกรมที่นำมาตรวจสอบว่ามีเส้นทางการทำงานใดบ้าง และในแต่ละเส้นทางมีผลของการทำงานอย่างไร และได้ผลการทำงานในแต่ละเส้นทางเป็นอย่างไร

ดังแสดงให้เห็นในรูปที่ 4.4



รูปที่ 4.4 รูปแสดงผลของเส้นทางการทำงานของโปรแกรมที่ได้จากการทดสอบ

และในการทดสอบประสิทธิภาพของ White-box Testing นั้น จะทำโดยการใส่ค่าช่วงของข้อมูลที่จะทำการทดสอบเข้าไป แล้วโปรแกรมจะนำเอาค่าในช่วงของข้อมูลเข้าไปวิเคราะห์การทำงานของโปรแกรมโดยจะนำค่ามาทำการรันเข้าไปดูว่าในส่วนใดของโปรแกรมได้มีการเข้าไปทำงานบ้างและในส่วนใดบ้างที่ไม่ได้ทำงานเลย ตัวโปรแกรมก็จะแสดงออกมาให้เห็นว่าที่เส้นทางใดบ้างที่ไม่สามารถเข้าไปทำงานได้เลยในช่วงของข้อมูลที่ได้กำหนดไว้ ดังแสดงให้เห็นในรูปที่ 4.5



รูปที่ 4.5 รูปแสดงผลของเส้นทางการทำงานของโปรแกรมที่ไม่ได้เข้าไปทำงาน

4.3 การทดสอบโปรแกรมโดยวิธี Black Box Testing

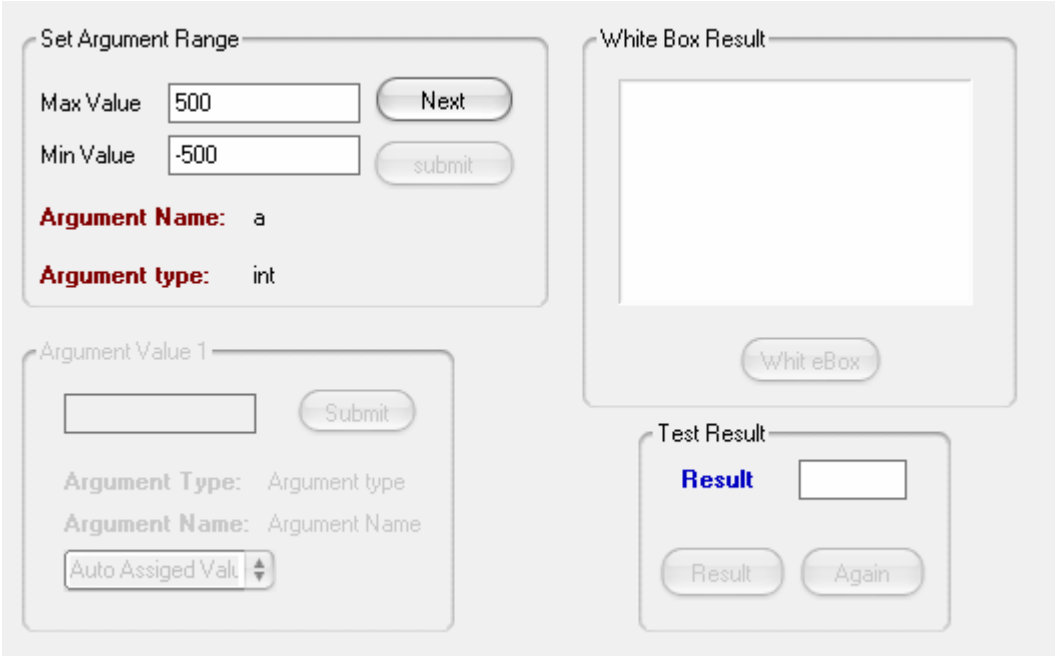
เป็นการสร้างกรณีทดสอบโดยพิจารณาจากข้อกำหนด (Specification) และเงื่อนไข (Condition) ของข้อมูลนั้น การทดสอบสนใจเฉพาะข้อมูลเข้าและผลลัพธ์ของโมดูลที่ต้องการทดสอบ เพื่อให้ตรงตามข้อกำหนดที่ได้ออกแบบไว้เท่านั้น ซึ่งจะไม่พิจารณาถึงโครงสร้างภายในโมดูลนั้น

ดังนั้นในการวิจัยทดลองนี้ สำหรับในส่วนของการทดสอบโดยวิธี Black-box Testing นั้นก็จะพิจารณาเพียง Argument ที่จะนำไปใช้เท่านั้น โดยที่ทางผู้ทำการทดสอบโปรแกรมจะต้องทำการเขียนโปรแกรมเพิ่มเพื่อบอกถึงว่ามีการใช้ค่า Argument ใดบ้าง และเป็นชนิดใดเพิ่มเข้าไปด้วย

เช่น โปรแกรมทดสอบ Test.C มีการใช้ค่า Argument a และ b เป็นชนิด integer ก็จะทำการประกาศเพิ่มไปคือ /*a a:int b:int a*/

4.4 การทดลองและผลการทดลองในส่วนของวิธี Black-box Testing

การทดลองนี้จะเป็นส่วนที่ต่อเนื่องมาจากการทดลองในส่วนของ White-box Testing ซึ่งได้ทำการคอมไพล์ไว้แล้ว เพราะการทดลองในช่วงนี้มีขอบเขตข้อกำหนดคือ ไฟล์ที่จะนำมาทดสอบนั้นจะต้องสามารถผ่านการคอมไพล์และแก้ไข Error มาเรียบร้อยแล้ว



รูปที่ 4.6 แสดงการตั้งค่าขอบเขตของ Argument

จากนั้นจะเป็นการป้อนค่าของ Argument ในค่าต่างๆ เพื่อนำไปทำการทดสอบผลลัพธ์ของโปรแกรมว่าค่าที่ได้จากการทดสอบนั้นมีข้อผิดพลาดเกิดขึ้นหรือไม่ ดังแสดงให้เห็นดังรูปที่ 4.7 ซึ่งสามารถทำการเลือกค่าเป็น

ค่าสูงสุด (Max)

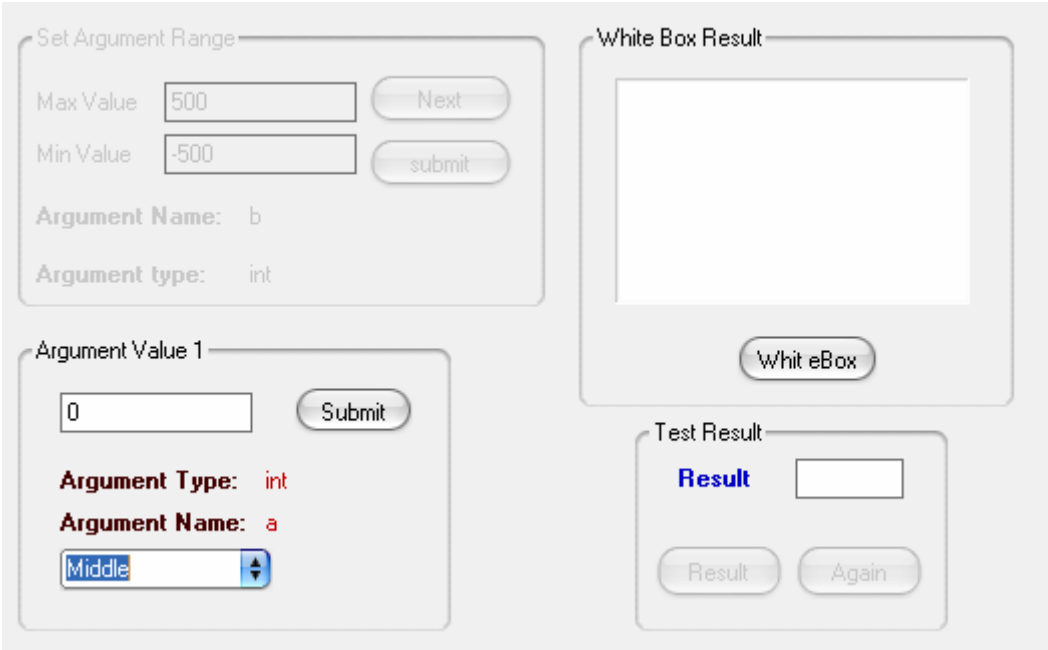
ค่าต่ำสุด (Min)

ค่ากึ่งกลางของขอบเขต (Middle)

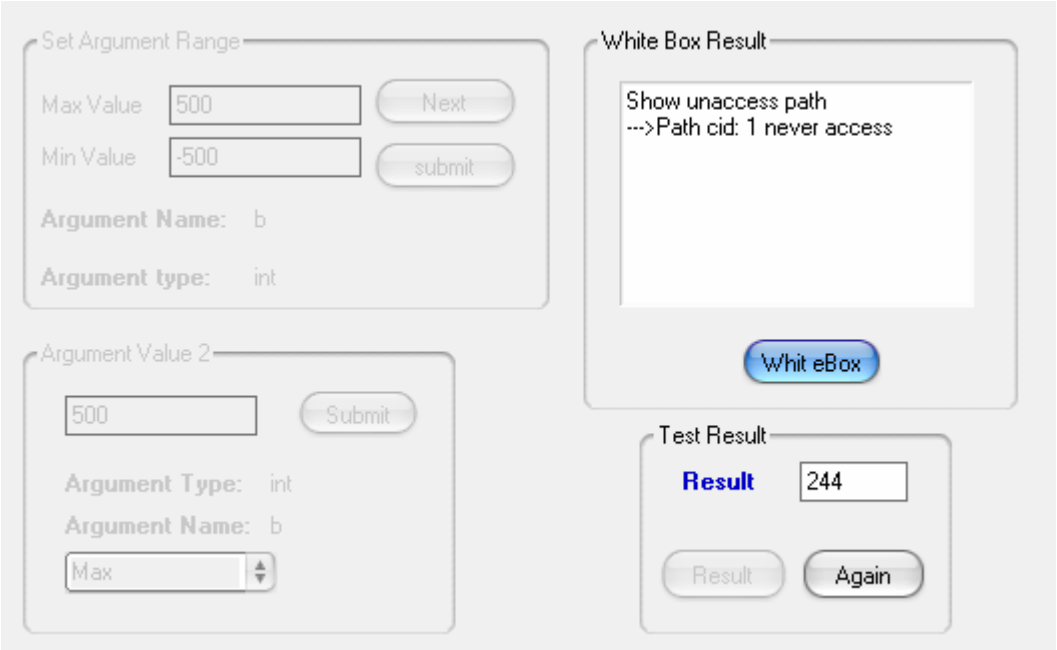
เลือกทำการสุ่มค่าออกมาก็ได้ (Random)

หรือจะใส่ค่าที่ต้องการทดสอบลงไปเลยก็ได้

และผลลัพธ์ที่ได้ของโปรแกรม ดังแสดงให้เห็นในรูปที่ 4.8



รูปที่ 4.7 แสดงการป้อนค่าให้กับ Argument เพื่อนำไปทดสอบ



รูปที่ 4.8 แสดงผลลัพธ์ของโปรแกรมที่ได้จากการทดสอบ