BÁO CÁO BÀI TẬP: KIỂM THỬ DÒNG DỮ LIỆU

Môn học: Kiểm thử và đảm bảo chất lượng phần mềm (INT3117)

Họ và tên sinh viên: Nguyễn Chí Trung
Mã sinh viên: 22028075

1 Bài 1

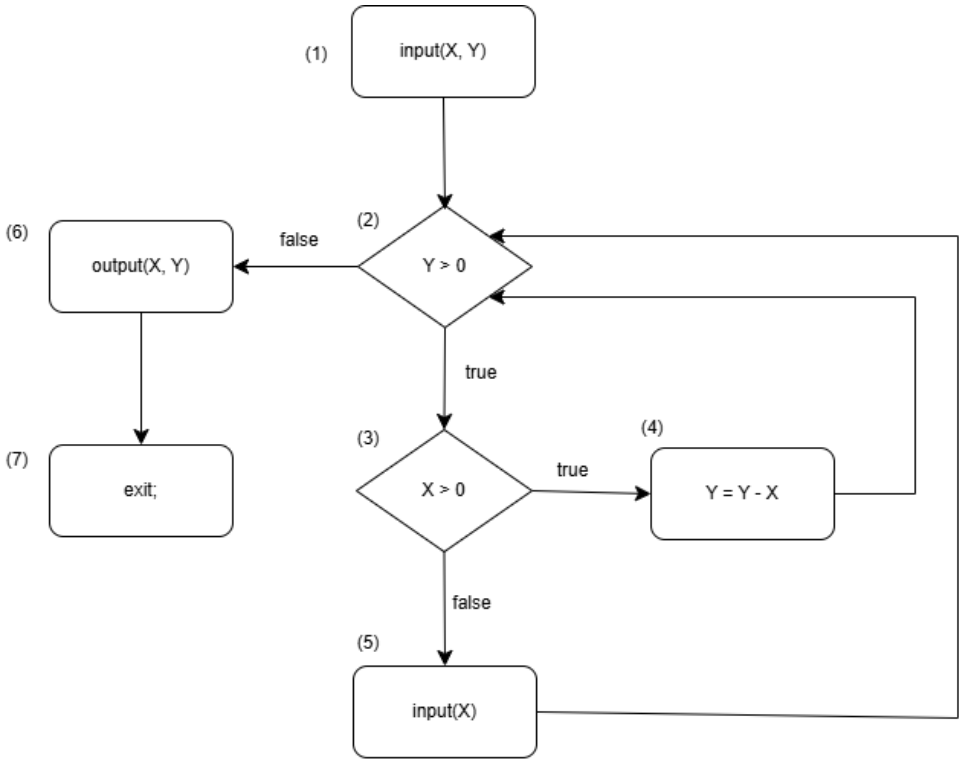
Trình bày các bước trong quy trình kiểm thử dòng dữ liệu động

Để thực hiện việc kiểm thử một chương trình theo quy trình kiểm thử dòng dữ liệu, ta cần thực hiện các bước sau:

- **Bước 1: Xác định các biến và luồng dữ liệu:** Trong bước này, thông qua việc phân tích mã nguồn, ta tìm được các biến xuất hiện trong chương trình, cũng như khi nào chúng được khai báo (def), khi nào chúng được sử dụng (use) với các dòng, khối lệnh tương ứng.
- **Bước 2: Vẽ đồ thị dòng dữ liệu của chương trình,** ngoài ra xác định vị trí các dòng, khối lệnh $def, use(p-use, c-use)$ của từng biến.
- **Bước 3: Lựa chọn một hoặc một số tiêu chí kiểm thử luồng dữ liệu**, tương ứng với yêu cầu của đề bài, chẳng hạn các độ phủ như $all-def, all-c-uses, all-p-uses, all-uses, all-du-pair, ...$
- **Bước 4: Xác định các đường đi trên đồ thị dòng dữ liệu thỏa mãn các tiêu chí kiểm thử đã chọn phía trên**
- **Bước 5: Thiết kế bộ dữ liệu kiểm thử:** Từ các tập đường đi ứng với đồ thị, ta giải kết hợp các biểu thức điều kiện để có được giá trị / điều kiện của giá trị đầu vào, và sau đó sinh các ca kiểm thử tương ứng với các đường đi đã xác định, cũng như tính toán kết quả mong đợi với các ca kiểm thử có đầu vào đã sinh.
- **Bước 6: Thực hiện kiểm thử, quan sát và đánh giá kết quả kiểm thử:** Chạy chương trình với các ca kiểm thử đã thiết kế, ghi nhận và quan sát các kết quả đầu ra của chương trình để kiểm tra xem chương trình có hoạt động đúng hay không, có phát sinh lỗi nào không, nhằm có những hành động phù hợp như tìm kiếm các lỗi liên quan.
  - Nếu phát hiện lỗi, tiến hành sửa các lỗi có liên quan và thực hiện lại tất cả các ca kiểm thử.

2 Bài 2

2.1 Đồ thị dòng điều khiển cho hàm $input(X, Y)$ (CFG)



Hình 1: Đồ thị dòng điều khiển cho hàm $input(X, Y)$

	Biến X	**Biến Y**
Def	1, 5	1, 4
P-use	3	2
C-use	4, 6	4, 6

Bảng 1: $def, p-use, c-use$ tương ứng với X và Y

2.2 Xác định các $du-pairs$ cho biến X và Y

Var	**Def**	**Du-pairs**
X	1	(1, 3), (1, 4), (1, 6)
	5	(5, 3), (5, 4), (5, 6)
Y	1	(1, 2), (1, 4), (1, 6)
	4	(4, 2), (4, 4), (4, 6)

Bảng 2: $Du-pairs$ cho biến X và Y

2.3 Sinh đường đi và các ca kiểm thử với độ đo $all-uses$

Var	Def	Du-pairs	Def-clear path	Complete path	Input	Output
X	1	(1, 3(T))	1, 2(T), 3(T)	1, 2(T), 3(T), 4, 2(F), 6, 7	x = 10, y = 10	x = 10, y = 0
		(1, 3(F))	1, 2(T), 3(F)	1, 2(T), 3(F), 5, 2(T), 3(T), 4, 2(F), 6, 7	x = -10 10, y = 10	x = 10, y = 0
		(1, 4)	1, 2(T), 3(T), 4	1, 2(T), 3(T), 4, 2(F), 6, 7	x = 10, y = 10	x = 10, y = 0
		(1, 6)	1, 2(F), 6	1, 2(F), 6, 7	x = 10, y = -10	x = 10, y = -10
	5	(5, 3(T))	5, 2(T), 3(T)	1, 2(T), 3(F), 5, 2(T), 3(T), 4, 2(F), 6, 7	x = -10 10, y = 10	x = 10, y = 0
		(5, 3(F))	5, 2(T), 3(F)	1, 2(T), 3(F), 5, 2(T), 3(F), 5, 2(T), 3(T), 4, 2(F), 6, 7	x = -10 -10 10, y = 10	x = 10, y = 0
		(5, 4)	5, 2(T), 3(T), 4	1, 2(T), 3(F), 5, 2(T), 3(T), 4, 2(F), 6, 7	x = -10 10, y = 10	x = 10, y = 0
		(5, 6)	5, 2(F), 6	1, 2(T), 3(F), 5, 2(T), 3(T), 4, 2(F), 6, 7	x = -10 10, y = 10	x = 10, y = 0
Y	1	(1, 2(T))	1, 2(T)	1, 2(T), 3(T), 4, 2(F), 6, 7	x = 10, y = 10	x = 10, y = 0
		(1, 2(F))	1, 2(F)	1, 2(F), 6, 7	x = 10, y = -10	x = 10, y = -10
		(1, 4)	1, 2(T), 3(T), 4	1, 2(T), 3(T), 4, 2(F), 6, 7	x = 10, y = 10	x = 10, y = 0
		(1, 6)	1, 2(F), 6	1, 2(F), 6, 7	x = 10, y = -10	x = 10, y = -10
	4	(4, 2(T))	4, 2(T)	1, 2(T), 3(T), 4, 2(T), 3(T), 4, 2(F), 6, 7	x = 10, y = 20	x = 10, y = 0
		(4, 2(F))	4, 2(F)	1, 2(T), 3(T), 4, 2(F), 6, 7	x = 10, y = 10	x = 10, y = 0
		(4, 4)	4, 2(T), 3(T), 4	1, 2(T), 3(T), 4, 2(T), 3(T), 4, 2(F), 6, 7	x = 10, y = 20	x = 10, y = 0
		(4, 6)	4, 2(F), 6	1, 2(T), 3(T), 4, 2(F), 6, 7	x = 10, y = 10	x = 10, y = 0

Bảng 3: Các đường đi ứng với ca kiểm thử

3 Bài 3

3.1 Các câu lệnh ứng với các khái niệm $def, c-use$ và $p-use$ tương ứng với các biến được sử dụng trong chương trình

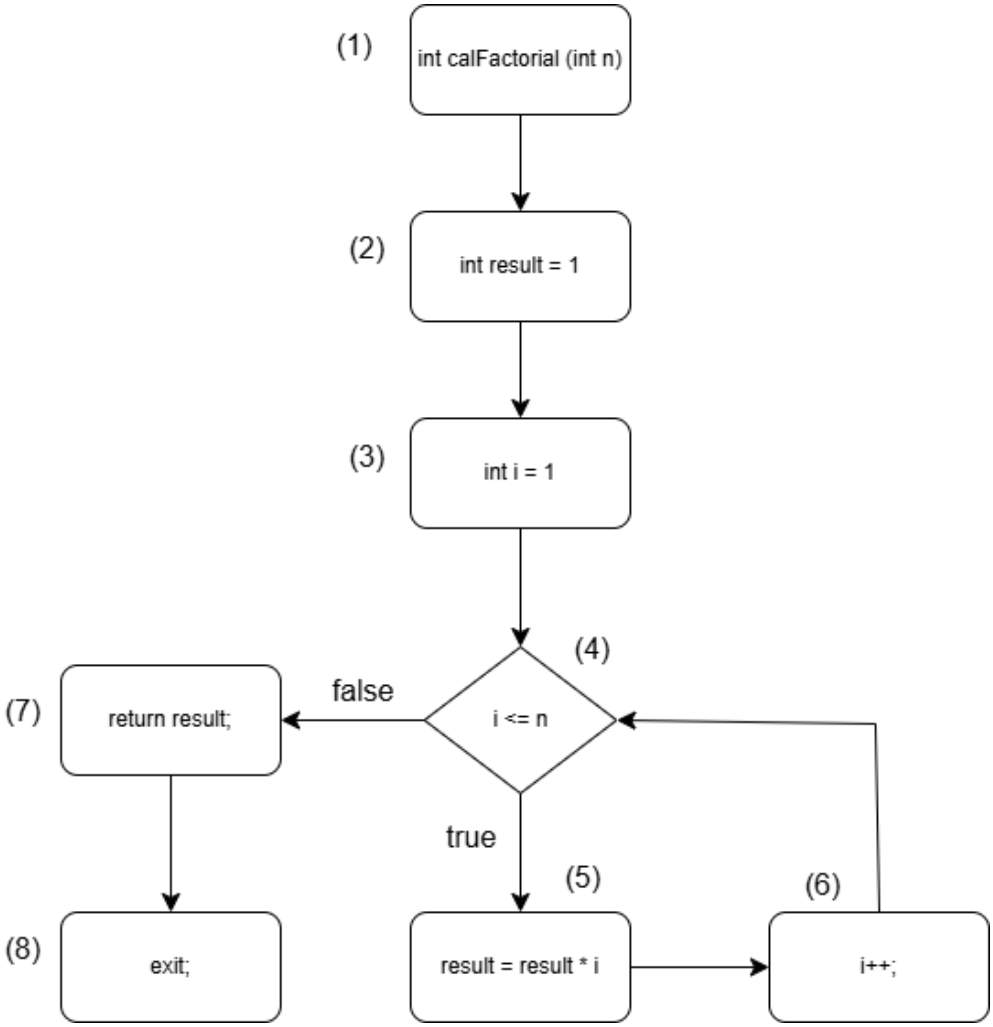
Để thuận tiện cho việc xác định các câu lệnh liên quan, các dòng lệnh trong chương trình sẽ được đánh số tương ứng như sau:

```
1 int calFactorial (int n) {
2     int result = 1;
3     int i=1;
4     while (i <= n){
5         result = result *i;
6         i++;
7     }//end while
8     return result;
9 }//the end
```

Var	Def	P-use	C-use
n	1	4	
$result$	2, 5		5, 8
i	3, 6	4	5, 6

Bảng 4: $Def, P-use, C-use$ ứng với biến $n, result, i$

3.2 Đồ thị dòng dữ liệu của hàm $calFactorial$



Hình 2: Đồ thị dòng dữ liệu của hàm $calFactorial$

4 Bài 4

4.1 Các $def - clear - path$ tương ứng với biến x và y

	Biến X	**Biến Y**
Def	0, 3	0, 2, 5
P-use	1, 4	1, 4
C-use	3, 5	6

Bảng 5: $def, p - use, c - use$ tương ứng với x và y

Var	Def	Def-clear path
x	0	0, 1
		0, 1, 2
		0, 1, 3
		0, 1, 2, 4
		0, 1, 2, 4, 5
		0, 1, 2, 4, 5, 6
		0, 1, 2, 4, 6
	3	3, 4
		3, 4, 5
		3, 4, 5, 6
		3, 4, 6
y	0	0, 1
		0, 1, 3
		0, 1, 3, 4
		0, 1, 3, 4, 6
	2	2, 4
		2, 4, 6
	5	5, 6

Bảng 6: $Def - clear\ path$ của biến x và y

4.2 Các $du - paths$ của biến x và y

Var	Du-pair	Du-path
x	(0, 1)	0, 1
	(0, 3)	0, 1, 3
	(0, 4)	0, 1, 2, 4
	(0, 5)	0, 1, 2, 4, 5
	(3, 4)	3, 4
	(3, 5)	3, 4, 5
y	(0, 1)	0, 1
	(0, 4)	0, 1, 3, 4
	(0, 6)	0, 1, 3, 4, 6
	(2, 4)	2, 4
	(2, 6)	2, 4, 6
	(5, 6)	5, 6

Bảng 7: $Du - paths$ của biến x và y

4.3 Các $All - p - uses/Some - c - uses$ và $All - c - uses/Some - p - uses$

Do có 2 trường hợp xảy ra (True / False) với các dòng / khối lệnh p-use, nên tương ứng với mỗi trường hợp, ta sẽ có các $def - clear$ path tương ứng:

Var	Du pair	Def-clear path	Complete path
x	(0, 1)	0, 1	0, 1, 2, 4, 6
			0, 1, 3, 4, 6
	(0, 4)	0, 1, 2, 4	0, 1, 2, 4, 6
			0, 1, 2, 4, 5, 6
	(3, 4)	3, 4	0, 1, 3, 4, 6
			0, 1, 3, 4, 5, 6
y	(0, 1)	0, 1	0, 1, 2, 4, 6
			0, 1, 3, 4, 6
	(0, 4)	0, 1, 3, 4	0, 1, 3, 4, 6
			0, 1, 3, 4, 5, 6
	(2, 4)	2, 4	0, 1, 2, 4, 6
			0, 1, 2, 4, 5, 6

Bảng 8: $All - p - uses/Some - c - usess$ của biến x và y

Var	Du pair	Def-clear path	Complete path
x	(0, 3)	0, 1, 3	0, 1, 3, 4, 6
	(0, 5)	0, 1, 2, 4, 5	0, 1, 2, 4, 5, 6
	(3, 5)	3, 4, 5	0, 1, 3, 4, 5, 6
y	(0, 6)	0, 1, 3, 4, 6	0, 1, 3, 4, 6
	(2, 6)	2, 4, 6	0, 1, 2, 4, 6
	(5, 6)	5, 6	0, 1, 3, 4, 5, 6

Bảng 9: $All - c - uses/Some - p - usess$ của biến x và y

4.4 Biểu thức của các $p-use(x, y)$ tại cạnh (1, 3) và (4, 5) lần lượt là $x+y = 4$ và $x^2 + y^2 > 17$. Giải thích đường đi (0 - 1 - 3 - 4 - 5 - 6) có thực thi được không

Ta có hệ phương trình: $\begin{cases} x + y = 4 \\ x^2 + y^2 > 17 \end{cases}$

Với phương trình $x + y = 4 => y = 4 - x$, thay vào bất phương trình còn lại ta được:

$$x^2 + (4 - x)^2 > 17 \tag{1}$$

$$2x^2 - 8x + 16 > 17 \tag{2}$$

$$x^2 - 4x + 8 > \frac{17}{2} \tag{3}$$

$$(x - 2)^2 > \frac{17}{2} - 4 = \frac{9}{2} \tag{4}$$

$$\tag{5}$$

$\Rightarrow x > 2 + \sqrt{4.5}$ và $y = 4 - x$ hoặc $x < 2 - \sqrt{4.5}$ và $y = 4 - x$

Do đó, có vô số cặp giá trị đầu vào (x, y) thực thi được đường đi $(0 - 1 - 3 - 4 - 5 - 6)$, ví dụ: $(5, -1)$. Vì vậy, đường đi (0 - 1 - 3 - 4 - 5 - 6) có thể thực thi được.

4.5 Giải thích lý do tại đỉnh 3 biến x được định nghĩa và sử dụng nhưng không tồn tại mối quan hệ $def - use$

Tại đỉnh 3 biến x được định nghĩa và sử dụng nhưng không tồn tại mối quan hệ $def - use$, ta sẽ dựa vào định nghĩa về mối quan hệ $def - use$ để giải thích. Đầu tiên, tại đỉnh 3 này, biến x được gán trả lại bằng một biểu thức có liên quan đến biến x, tức là $use(x)$ sẽ được sử dụng trước $def(x)$, nhưng tuy nhiên, độ dài từ đỉnh này tới chính nó là 0, và không có đường đi quay lại (đồ thị tuần tự) nên sẽ không tồn tại 1 $def - clear\ path$ tương ứng. Do đó, đỉnh 3 biến x không tồn tại mối quan hệ $def - use$.

5 Bài 5

5.1 Xây dựng CFG cho hàm UCLN với đồ thị C_2

5.2 Sinh đường đi và các ca kiểm thử với độ đo C_2

5.3 Sinh đường đi và các ca kiểm thử với độ đo $all - def - coverage$

6 Bài 6

6.1 Mô tả bài toán

Một công ty thuộc lĩnh vực vận chuyển hàng hóa tính phí giao hàng dựa trên khoảng cách (x), trọng lượng của hàng hóa (y) và thời gian giao hàng mong muốn (z). Hãy tính tổng số tiền khách hàng phải trả cho phí vận chuyển, với các điều kiện sau:

- Với khoảng cách (km):
  - Dưới 10km: 5.000 VNĐ/km
  - Từ 10km đến dưới 50km: 4.500 VNĐ/km
  - Từ 50km trở lên: 4.000 VNĐ/km
- Với trọng lượng của hàng hóa (kg):
  - Dưới 10kg: Không thu thêm phí
  - Từ 10kg trở lên: 50.000 VNĐ
- Với thời gian giao hàng mong muốn (h):
  - Dưới 12h: Phụ phí 50.000 VNĐ
  - Từ 12h trở lên: Không thu thêm phí

Đầu vào: Ba số x, y, z với $x, y \in R, z \in Z, 0 < x \leq 1000, 0 < y \leq 100, 0 < z \leq 24$
(x, y làm tròn đến 2 chữ số sau dấu thập phân)

Đầu ra: Số tiền mà khách hàng phải trả.

Link đến mã nguồn trên Github: Kiểm thử dòng dữ liệu

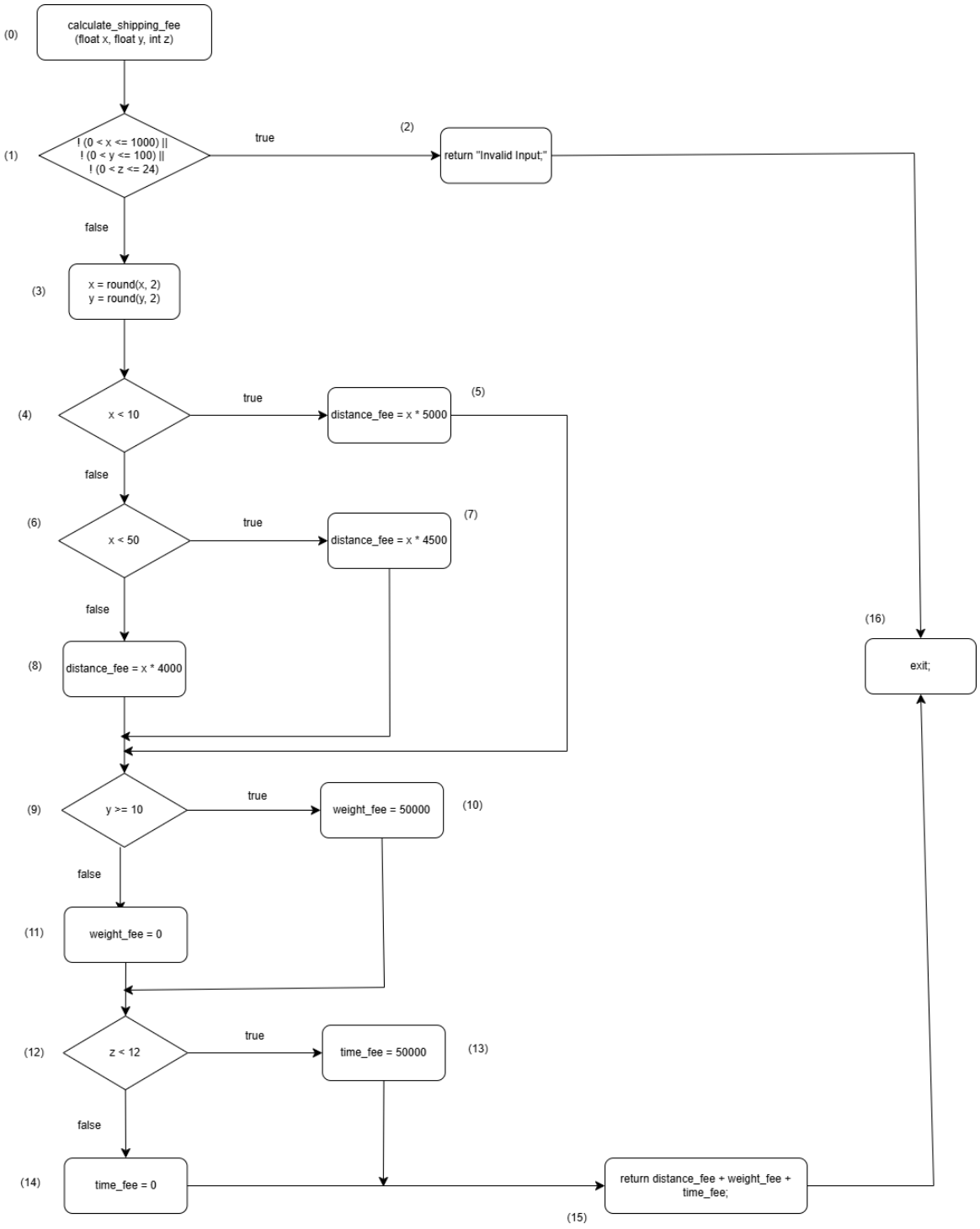
6.2 Chương trình của bài toán

```python
def calculate_shipping_fee(x: float, y: float, z: int) -> float:
    if not (0 < x <= 1000) or not (0 < y <= 100) or not (0 < z <= 24):
        return "Invalid Input"

    x, y = round(x, 2), round(y, 2)
    distance_fee = x * 5000 if x < 10 else (x * 4500 if x < 50 else x * 4000)
    weight_fee = 50000 if y >= 10 else 0
    time_fee = 50000 if z < 12 else 0

    return distance_fee + weight_fee + time_fee
```

6.3 Đồ thị dòng điều khiển của hàm *calculate_shipping_fee*



Hình 3: Đồ thị dòng điều khiển cho hàm *calculate_shipping_fee*

6.4 Kiểm thử với độ phủ $all - uses$

6.4.1 Xác định các khái niệm $def, c-use$ và $p-use$ tương ứng với các biến được sử dụng trong chương trình

	Biến x	Biến y	Biến z
Def	0, 3	0, 3	0
P-use	1, 4, 6	1, 9	1, 12
C-use	3, 5, 7, 8	3	

Bảng 10: $def, p-use, c-use$ tương ứng với x, y và z

6.4.2 Sinh đường đi và các ca kiểm thử với độ đo $all - uses$

Var	Def	Du-pairs	Def-clear path	Complete path	Input	Output
x	0	(0, 1(T))	0, 1(T)	0, 1(T), 2, 16	x = -15.42, y = 10, z = 12	Invalid Input
		(0, 1(F))	0, 1(F)	0, 1(F), 3, 4(T), 5, 9(T), 10, 12(T), 13, 15, 16	x = 9.08, y = 12.4, z = 11	145400
		(0, 3)	0, 1(F), 3	0, 1(F), 3, 4(T), 5, 9(T), 10, 12(T), 13, 15, 16	x = 6.43, y = 15, z = 4	132150
	3	(3, 4(T))	3, 4(T)	0, 1(F), 3, 4(T), 5, 9(T), 10, 12(T), 13, 15, 16	x = 9.08, y = 12.4, z = 11	145400
		(3, 4(F))	3, 4(F)	0, 1(F), 3, 4(F), 6(T), 7, 9(F), 11, 12(F), 14, 15, 16	x = 40.05, y = 8.75, z = 18	180225
		(3, 6(T))	3, 4(F), 6(T)	0, 1(F), 3, 4(F), 6(T), 7, 9(F), 11, 12(F), 14, 15, 16	x = 25, y = 5.13, z = 20	112500
		(3, 6(F))	3, 4(F), 6(F)	0, 1(F), 3, 4(F), 6(F), 8, 9(F), 11, 12(T), 13, 15, 16	x = 125.25, y = 6.12, z = 9	551000
		(3, 5)	3, 4(T), 5	0, 1(F), 3, 4(T), 5, 9(T), 10, 12(T), 13, 15, 16	x = 9.08, y = 12.4, z = 11	145400
		(3, 7)	3, 4(F), 6(T), 7	0, 1(F), 3, 4(F), 6(T), 7, 9(F), 11, 12(F), 14, 15, 16	x = 40.05, y = 8.75, z = 18	180225
		(3, 8)	3, 4(F), 6(F), 8	0, 1(F), 3, 4(F), 6(F), 8, 9(F), 11, 12(T), 13, 15, 16	x = 68.42, y = 3.12, z = 6	323680
y	0	(0, 1(T))	0, 1(T)	0, 1(T), 2, 16	x = 100.5, y = 100.01, z = 12	Invalid Input
		(0, 1(F))	0, 1(F)	0, 1(F), 3, 4(T), 5, 9(T), 10, 12(T), 13, 15, 16	x = 9.08, y = 12.4, z = 11	145400
		(0, 3)	0, 1(F), 3	0, 1(F), 3, 4(T), 5, 9(T), 10, 12(T), 13, 15, 16	x = 9.08, y = 12.4, z = 11	145400
	3	(3, 9(T))	3, 4(T), 5, 9(T)	0, 1(F), 3, 4(T), 5, 9(T), 10, 12(T), 13, 15, 16	x = 9.08, y = 12.4, z = 11	145400
		(3, 9(F))	3, 4(F), 6(T), 7, 9(F)	0, 1(F), 3, 4(F), 6(T), 7, 9(F), 11, 12(F), 14, 15, 16	x = 40.05, y = 8.75, z = 18	180225
z	0	(0, 1(T))	0, 1(T)	0, 1(T), 2, 16	x = 52.36, y = 10, z = -1	Invalid Input
		(0, 1(F))	0, 1(F)	0, 1(F), 3, 4(T), 5, 9(T), 10, 12(T), 13, 15, 16	x = 9.08, y = 12.4, z = 11	145400
		(0, 12(T))	0, 1(F), 3, 4(T), 5, 9(T), 10, 12(T)	0, 1(F), 3, 4(T), 5, 9(T), 10, 12(T), 13, 15, 16	x = 9.08, y = 12.4, z = 11	145400
		(0, 12(F))	0, 1(F), 3, 4(F), 6(T), 7, 9(F), 11, 12(F)	0, 1(F), 3, 4(F), 6(T), 7, 9(F), 11, 12(F), 14, 15, 16	x = 40.05, y = 8.75, z = 18	180225

Bảng 11: Các đường đi ứng với ca kiểm thử

6.5 Tiến hành kiểm thử

6.5.1 Chương trình kiểm thử cho bài toán

```
import unittest
from calculate_shipping_fee import calculate_shipping_fee

class TestShippingFee(unittest.TestCase):
    def test_cases_branch_coverage(self):
        test_data = [
            (-15.42, 10, 12, "Invalid Input"),
```

```
            (100.5, 100.01, 12, "Invalid Input"),
            (52.36, 10, -1, "Invalid Input"),
            (9.08, 12.4, 11, 145400),
            (6.43, 15, 4, 132150),
            (40.05, 8.75, 18, 180225),
            (25, 5.13, 20, 112500),
            (125.25, 6.12, 9, 551000),
            (68.42, 3.12, 6, 323680)
        ]

        for x, y, z, expected in test_data:
            with self.subTest(x=x, y=y, z=z):
                self.assertEqual(calculate_shipping_fee(x, y, z), expected)

if __name__ == "__main__":
    unittest.main()
```

6.5.2 Kết quả kiểm thử

STT	Input	Expected Output	Actual Output	Result
1	x = -15.42, y = 10, z = 12	Invalid Input	Invalid Input	Pass
2	x = 100.5, y = 100.01, z = 12	Invalid Input	Invalid Input	Pass
3	x = 52.36, y = 10, z = -1	Invalid Input	Invalid Input	Pass
4	x = 9.08, y = 12.4, z = 11	145400	145400	Pass
5	x = 6.43, y = 15, z = 4	132150	132150	Pass
6	x = 40.05, y = 8.75, z = 18	180225	180225	Pass
7	x = 25, y = 5.13, z = 20	112500	112500	Pass
8	x = 125.25, y = 6.12, z = 9	551000	551000	Pass
9-	x = 68.42, y = 3.12, z = 6	323680	323680	Pass

Bảng 12: Kết quả khi kiểm thử với bộ test case