ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

BÁO CÁO
BÀI TẬP

Sinh viên: Nguyễn Đức Duy
Mã sinh viên: 22028215
Lớp: INT3117_1

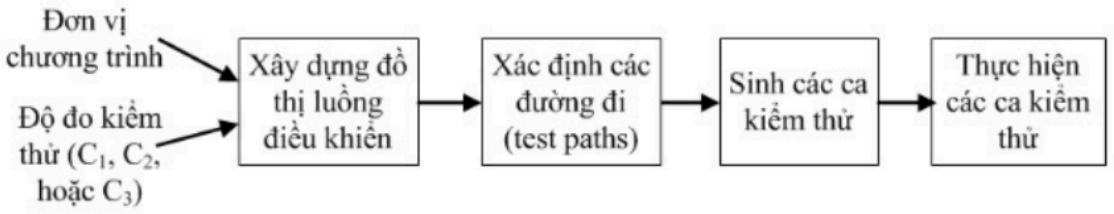
1. Trình bày các bước nhằm kiểm thử một đơn vị chương trình theo phương pháp kiểm thử dòng điều khiển với một độ đo kiểm thử cho trước.

Bài làm:

- **Bước 1:** Xác định đơn vị chương trình cần kiểm thử (Chọn module hoặc hàm cụ thể)
- **Bước 2:** Phân tích mã nguồn và xây dựng đồ thị dòng điều khiển (Control Flow Graph – CFG)
    - Mỗi nút (node) đại diện cho một câu lệnh hoặc khối lệnh.

    - Mỗi cạnh (edge) thể hiện luồng điều khiển giữa các khối (ví dụ: sau `if`, chương trình có thể đi nhánh true hoặc false).
- **Bước 3:** Xác định độ đo kiểm thử cần đạt
    - Phủ cấp 1 (phủ câu lệnh): Mỗi câu lệnh trong chương trình được thực thi ít nhất một lần sau khi chạy các ca kiểm thử
    - Phủ cấp 2 (phủ nhánh): Lựa chọn các đường đi sao cho mỗi nhánh đều được đi qua ít nhất một lần
    - Phủ cấp 3 (phủ điều kiện): Kiểm thử sao cho mỗi điều kiện con của từng điểm quyết định đều được thực hiện ít nhất một lần cho trường hợp True và False
    - Bao phủ vòng lặp

- **Bước 4**: Thiết kế bộ test case dựa trên CFG và độ đo đã chọn: Dựa vào luồng điều khiển, ta chọn giá trị đầu vào sao cho đảm bảo bao phủ được các nhánh / câu lệnh / đường đi tương ứng.
- **Bước 5**: Thực thi các test case và ghi nhận kết quả
- **Bước 6**: Đánh giá mức độ bao phủ và phân tích kết quả



Hình 6.4: Quy trình kiểm thử đơn vị chương trình dựa trên độ đo.

2.

9. Cho hàm được viết bằng ngôn ngữ C như Đoạn mã 6.1.

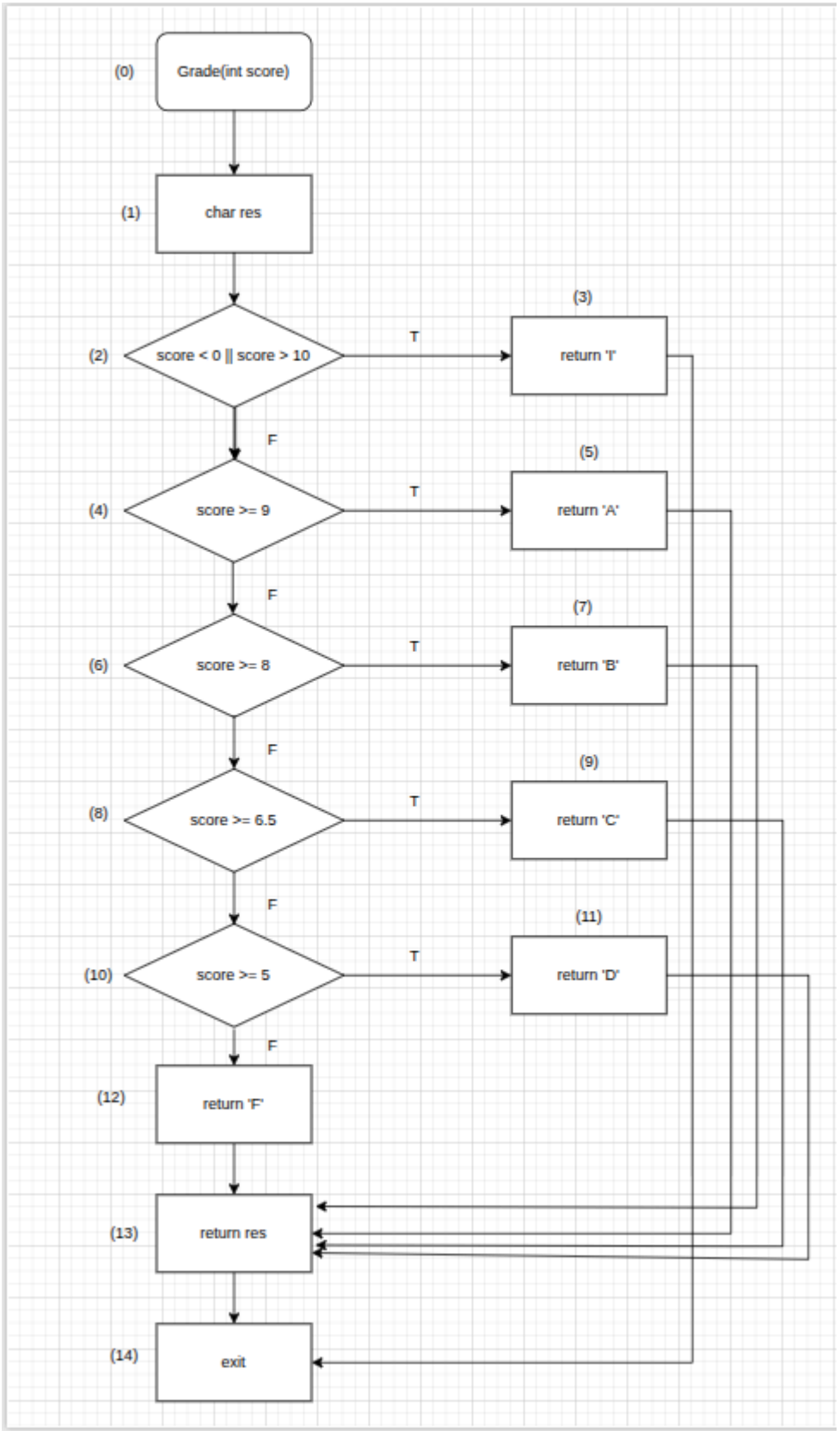
Đoạn mã 6.1: Mã nguồn của hàm Grade

```
char Grade(int score){

      int res;
      if(score < 0 || score > 10)
        return ''I;
      if(score>=9)
        res = 'A';
      else
        if(score >=8)
              res ='B';
        else
              if(score >=6.5)
                    res = 'C';
              else
                    if(score >=5)
                       res = 'D';
                    else
                       res = 'F';
      return res;
}
```

- Hãy xây dựng đồ thị dòng điều khiển cho hàm **Grade** ứng với độ đo C_1 và C_2.
- Hãy sinh các đường đi và các ca kiểm thử với độ đo C_1.
- Hãy sinh các đường đi và các ca kiểm thử với độ đo C_2.

a. Xây dựng đồ thị dòng điều khiển cho hàm Grade ứng với độ đo C1 và C2



b. Các đường đi và các ca kiểm thử với độ đo C1

Path 1: 0 -> 1 -> 2(T) -> 3 -> 14

=> Test case: Grade(11)

Path 2: 0 -> 1 -> 2(F) -> 4(T) -> 5 -> 13 -> 14

=> Test case: Grade(9)

Path 3: 0 -> 1 -> 2(F) -> 4(F) -> 6(T) -> 7 -> 13 -> 14

=> Test case: Grade(8)

Path 4: 0 -> 1 -> 2(F) -> 4(F) -> 6(F) -> 8(T) -> 9 -> 13 -> 14

=> Test case: Grade(6.5)

Path 5: 0 -> 1 -> 2(F) -> 4(F) -> 6(F) -> 8(F) -> 10(T) -> 11 -> 13 -> 14

=> Test case: Grade(5)

Path 6: 0 -> 1 -> 2(F) -> 4(F) -> 6(F) -> 8(F) -> 10(F) -> 12 -> 13 -> 14

=> Test case: Grade(3)

Scov = 15/15

c. Các đường đi và các ca kiểm thử với độ đo C2

Path 1: 0 -> 1 -> 2(T) -> 3 -> 14

=> Test case: Grade(11)

Path 2: 0 -> 1 -> 2(F) -> 4(T) -> 5 -> 13 -> 14

=> Test case: Grade(9)

Path 3: 0 -> 1 -> 2(F) -> 4(F) -> 6(T) -> 7 -> 13 -> 14

=> Test case: Grade(8)

Path 4: 0 -> 1 -> 2(F) -> 4(F) -> 6(F) -> 8(T) -> 9 -> 13 -> 14

=> Test case: Grade(6.5)

Path 5: 0 -> 1 -> 2(F) -> 4(F) -> 6(F) -> 8(F) -> 10(T) -> 11 -> 13 -> 14

=> Test case: Grade(5)

Path 6: 0 -> 1 -> 2(F) -> 4(F) -> 6(F) -> 8(F) -> 10(F) -> 12 -> 13 -> 14

=> Test case: Grade(3)

Bcov = 10/10

3.

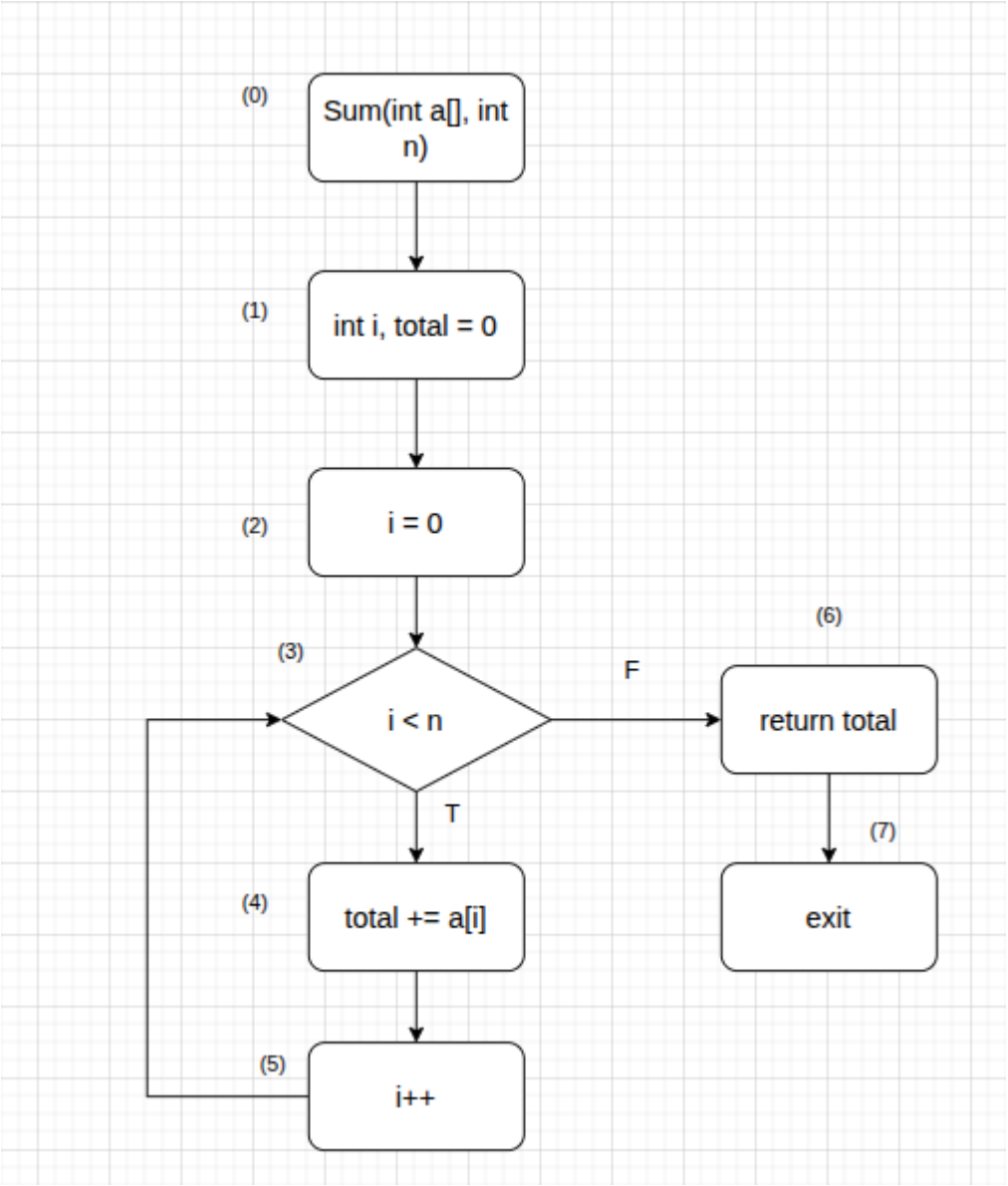
13. Cho hàm được viết bằng ngôn ngữ C như Đoạn mã 6.5.

Đoạn mã 6.5: Mã nguồn của hàm Sum

```
int Sum(int a[], int n){
        int i, total = 0;
        for(i=0; i<n; i++)
                total = total + a[i];
        return total;
}
```

- Hãy xây dựng đồ thị dòng điều khiển cho hàm `Sum` ứng với độ đo C_1 và C_2.
- Hãy sinh các đường đi và các ca kiểm thử với độ đo C_1.
- Hãy sinh các đường đi và các ca kiểm thử với độ đo C_2.
- Hãy sinh các ca kiểm thử để kiểm thử vòng lặp `for`.

a.



b.

Path 1: 0 -> 1 -> 2 -> 3(T) -> 4 -> 5 -> 3(F) -> (6) -> (7)
=> Test case: Sum(a, 1)

Scov = 8/8

c.

Path 1: 0 -> 1 -> 2 -> 3(F) -> (6) -> (7)

=> Test case: Sum(a, 0)

Path 2: 0 -> 1 -> 2 -> 3(T) -> 4 -> 5 -> 3(F) -> (6) -> (7)

=> Test case: Sum(a, 1)

Bcov = 2/2

d.

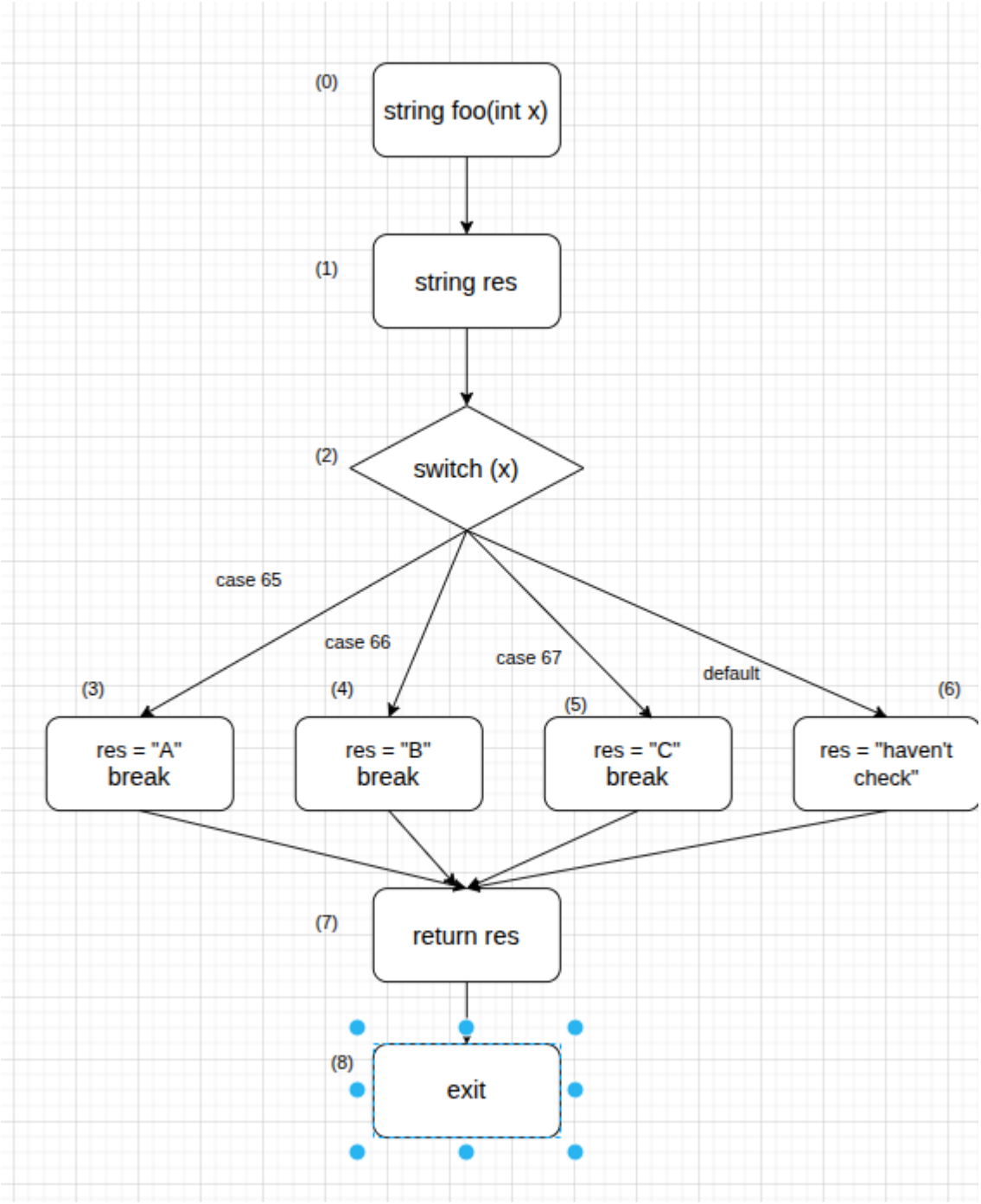
Số lần lặp	Test case
0	Sum(a, 0)
1	Sum(a, 1)
2	Sum(a, 2)
n	Sum(a, n)
n + 1	Không thể sinh

4.

```
string foo(int x){
        string res;
        switch(x):
                case 65:
                        res = "A";
                        break;
                case 66:
                        res = "B";
                        break;
                case 67:
                        res = "C";
                        break;
                default:
                        res = "haven't check";
        return res;
}
```

- Xây dựng đồ thị dòng điều khiển cho hàm foo ứng với độ đo C2
- Hãy sinh các đường đi và các ca kiểm thử ứng với độ đo C2

a.



b.

Path 1: 0 -> 1 -> 2(case 65) -> 3 -> 7 -> 8

Test case: foo(65)

Path 2: 0 -> 1 -> 2(case 66) -> 4 -> 7 -> 8

Test case: foo(66)

Path 3: 0 -> 1 -> 2(case 67) -> 5 -> 7 -> 8

Test case: foo(67)

Path 4: 0 -> 1 -> 2(default) -> 6 -> 7 -> 8

Test case: foo(5)

Bcov = 4/4

5. Link mã nguồn

5.1. Mô tả bài toán.

Hệ thống tính giá vé xem phim cho 1 người dựa vào Tuổi (Age) và Loại ngày (DayType)

- Nếu Age không hợp lệ (âm hoặc >100) -> báo lỗi
- Ngày chia 2 loại: ngày thường hoặc cuối tuần
- Nhóm tuổi:
    - Child: 0-12
    - Adult: 13-60
    - Senior: 60-100

Bảng giá:

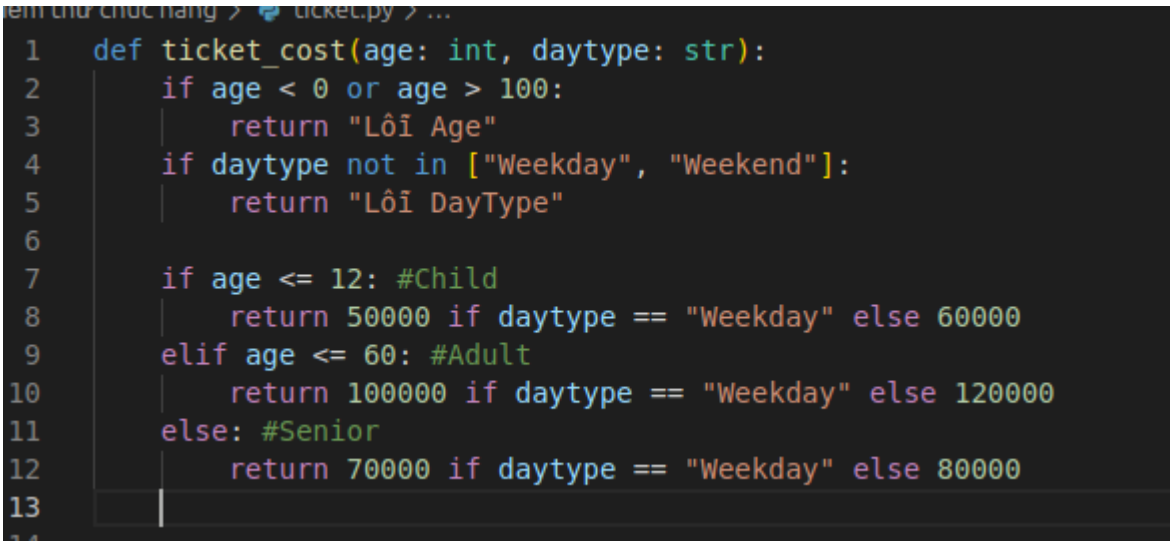
- Child: Ngày thường = 50,000 VND, Cuối tuần = 60,000 VND
- Adult: Ngày thường = 100,000 VND, Cuối tuần = 120,000 VND
- Senior: Ngày thường = 70,000 VND, Cuối tuần = 80,000 VND

Đầu vào: Một số nguyên x, với x ∈ [0..100] và một string là

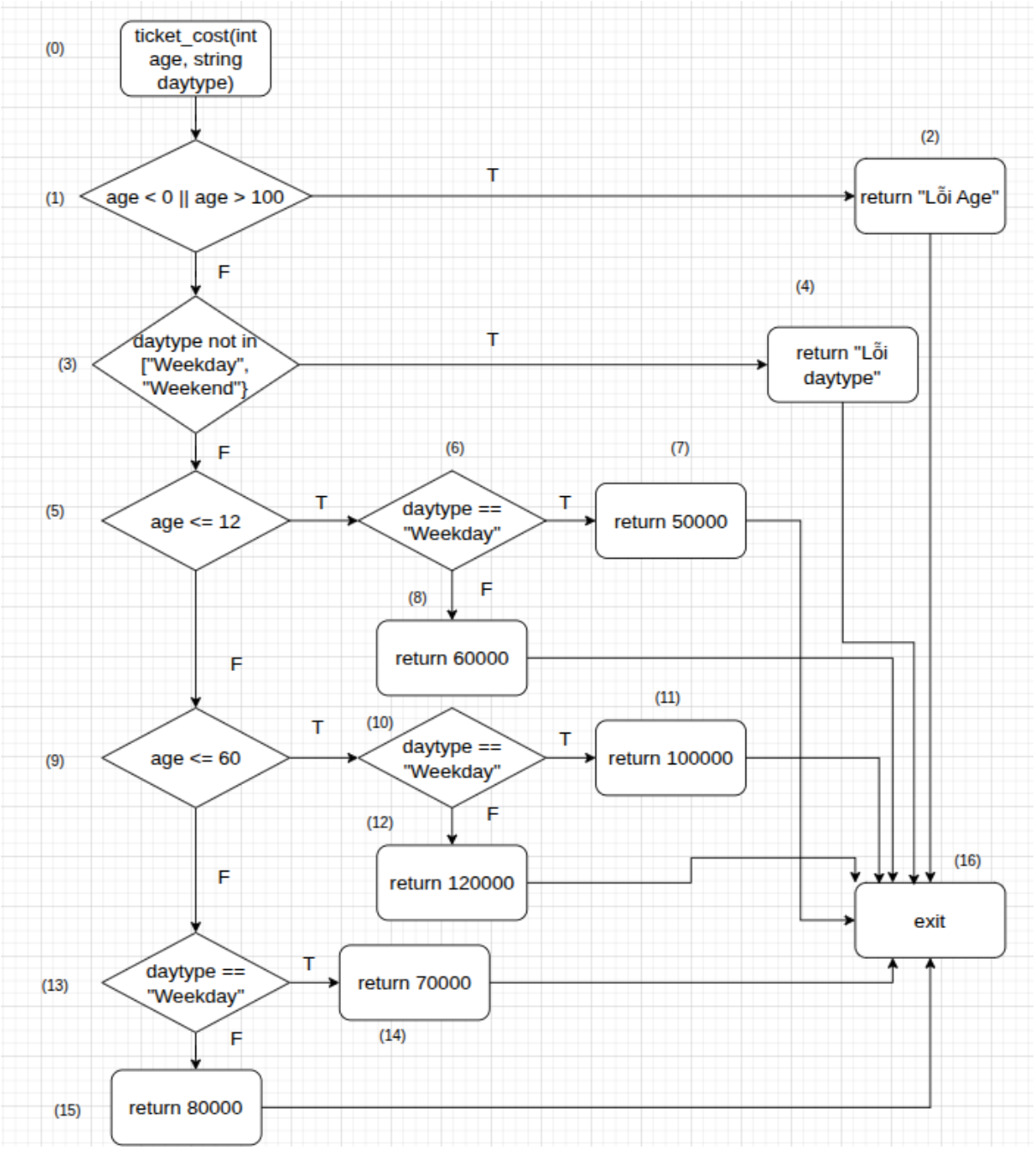
“Weekday” hoặc “Weekend”

Đầu ra: Giá vé hoặc báo lỗi.

5.2. Cài đặt chương trình



5.3 Xây dựng đồ thị dòng điều khiển cho hàm ticket_cost với độ phủ C2



5.4 Kiểm thử với độ phủ C2

STT	Path	Test case
1	0 -> 1(T)->2->16	ticket_cost(-5,”Weekday”)
2	0->1(F)->3(T)->4->16	ticket_cost(5,”deekday”)
3	0->1(F)->3(F)->5(T)->6(T)->7->16	ticket_cost(5,”Weekday”)

4	0->1(F)->3(F)->5(T)->6(F)->8->16	ticket_cost(5,”Weekend)
5	0->1(F)->3(F)->5(F)->9(T)->10(T)->11->16	ticket_cost(18,”Weekday”)
6	0->1(F)->3(F)->5(F)->9(T)->10(F)->12->16	ticket_cost(18,”Weekend”)
7	0->1(F)->3(F)->5(F)->9(F)->13(T)->14->16	ticket_cost(80,”Weekday”)
8	0->1(F)->3(F)->5(F)->9(F)->13(F)->15->16	ticket_cost(80,”Weekend”)

Bcov = 14/14

5.5 Kết quả kiểm thử bằng pytest

TC ID	Age	Daytype	EO	Result
1	-5	Weekday	Invalid Age	PASS
2	5	hello	Invalid Daytype	PASS
3	5	Weekday	50000	PASS
4	5	Weekend	60000	PASS
5	18	Weekday	100000	PASS
6	18	Weekend	120000	PASS
7	80	Weekday	70000	PASS
8	80	Weekend	80000	PASS