**1.1**

1/

*a += 4 (Tương đương với a = a + 4): Biến a ban đầu là 2. Nó được tăng thêm 4, nên a trở thành 6.*

*b = b - a Biến b ban đầu là 0. Nó trừ đi giá trị mới nhất của a (là 6). Kết quả: , nên b trở thành -6.*

*c += a + b (Tương đương với c = c + (a + b)):*

* *a + b là 6+(-6)=0*
* *c ban đầu là 2.*
* *c được cộng với 0. Kết quả: , nên c vẫn là 2.*

*d: Biến d không được nhắc đến trong bất kỳ phép toán nào, nên giá trị của nó giữ nguyên là 5.*

2/

*Lý do đầu ra của câu lệnh cout không xác định (Undefined Behavior) là vì nó sửa đổi (++a, a++) và truy cập (a) cùng một biến (a) nhiều lần trong cùng một câu lệnh mà không có điểm trình tự đảm bảo thứ tự đánh giá.*

*Tuy nhiên, nếu phải giải thích theo quy ước học thuật giả định thứ tự từ trái sang phải:*

1. *a (trước tiên): In ra giá trị hiện tại là 2.*
2. *++a: Tăng a lên trước (từ 2 thành 3), sau đó in ra 3.*
3. *a++: In ra giá trị hiện tại là 3, sau đó mới tăng a lên sau (từ 3 thành 4).*

*Kết quả in ra (theo giả định): 2 3 3*

*Giá trị cuối cùng của a:4*

**1.2**

*Kết quả của không tương đương giữa tất cả các dòng 1,2,4*

***Giải thích:***

*Sự khác biệt về kết quả là do sự khác nhau trong thời điểm các toán tử tăng/giảm (increment/decrement) thay đổi giá trị của biến:*

1. *Toán tử Tiền tố (++a, --a): Biến được cập nhật trước khi giá trị của nó được sử dụng trong biểu thức. (Ví dụ: Dòng 2, và được tính là 3 và 1).*
2. *Toán tử Hậu tố (a++, a--): Biến được cập nhật sau khi giá trị ban đầu của nó được sử dụng trong biểu thức. (Ví dụ: Dòng 3, và vẫn được tính là 2 và 0, sau đó mới giảm).*

*Việc kết hợp các toán tử này (như trong Dòng 4) tạo ra các giá trị đầu vào khác nhau cho phép cộng, dẫn đến kết quả cuối cùng khác nhau cho .*

**1.3**

1/

*Dự đoán: 2*

***Giải thích***

*Biểu thức c = ++a \* b-- + c++; gây ra* ***hành vi không xác định (undefined behavior)*** *trong C++, vì biến c vừa được* ***thay đổi*** *(c++) vừa* ***được gán giá trị*** *trong cùng một biểu thức mà không có thứ tự đánh giá rõ ràng.*

*Nếu* ***giả sử*** *trình biên dịch đánh giá* ***từ trái sang phải****, ta có thể mô phỏng như sau:*

1. *++a dùng 3 (a thành 3)*
2. *b-- dùng 0 (b thành -1)*
3. *c++ dùng 2 (c thành 3)*
4. *Phép tính: 3 × 0 + 2 = 2*
5. *c được gán 2*

*→ Kết quả giả định:* ***a = 3, b = -1, c = 2***

2/

*Dự đoán:1*

***Giải thích***

*Biểu thức d = a++ \* ++b - --c; gây ra* ***hành vi không xác định (undefined behavior)*** *trong C/C++, vì các biến a, b, c đều bị thay đổi và truy cập trong cùng một biểu thức mà không có thứ tự đánh giá rõ ràng.*

*Nếu giả sử trình biên dịch đánh giá* ***từ trái sang phải****, ta có thể mô phỏng như sau:*

* *a++ dùng 2, rồi a thành 3*
* *++b thành 1 và dùng 1*
* *--c thành 1 và dùng 1*
* *→ Kết quả: 2 \* 1 - 1 = 1, nên d = 1*

**1.4**

1/

*Dòng 1: Biểu thức này gây hành vi không xác định (undefined behavior) vì biến a bị sửa đổi hai lần trong cùng một điểm trình tự — một lần bởi a++ và một lần bởi toán tử -- áp dụng lên giá trị tạm thời trả về từ a++. Ngoài ra, không thể áp dụng toán tử giảm (--) cho giá trị tạm thời (rvalue), nên biểu thức này không hợp lệ theo chuẩn C++.*

*Dòng 2: Tương tự, biểu thức sửa đổi a hai lần (thông qua a++ và ++(...)) trong cùng một biểu thức mà không có thứ tự đánh giá xác định, dẫn đến hành vi không xác định. Cụ thể, việc dùng toán tử tăng tiền tố ++ lên kết quả của toán tử tăng hậu tố a++ là không hợp lệ và không được chuẩn C++ cho phép.*

2/

***Không***

***Giải thích:***

*Biểu thức này KHÔNG được định nghĩa rõ ràng (Not well-defined) vì nó sửa đổi biến a (a++) và truy cập nó nhiều lần trong cùng một biểu thức con (subexpression) mà không có điểm trình tự (sequence point) ở giữa. Trình biên dịch không biết nên thực hiện a++ nào trước, dẫn đến kết quả khác nhau tùy thuộc vào trình biên dịch.*

3/

*Lỗi biên dịch*

***Giải thích****:*

*Biến a được khai báo là const (hằng số), nghĩa là giá trị của nó không thể bị thay đổi sau khi khởi tạo. Toán tử tăng (a++) cố gắng sửa đổi giá trị này, điều này bị C++ ngăn chặn, dẫn đến lỗi biên dịch.*

4/

*Biểu thức gây hành vi không xác định (UB) trong C++ vì c bị sửa đổi hai lần trong cùng một biểu thức mà không có điểm trình tự giữa hai lần sửa đổi.*

*Hai khả năng (ví dụ giả định) cho kết quả:*

* *Nếu (c += 3) chạy trước: c từ 0 → 3, rồi (c += 4) → 7 ⇒ d = 3 × 7 = 21.*
* *Nếu (c += 4) chạy trước: c từ 0 → 4, rồi (c += 3) → 7 ⇒ d = 4 × 7 = 28.*

*Một số trình biên dịch trả 21 hoặc 28, nhưng chuẩn không đảm bảo thứ tự đánh giá; do đó kết quả là không xác định.*