Câu 1: Nêu khái niệm kiểm thử hộp đen. Cho 1 ví dụ đơn giản về kiểm thử hộp đen

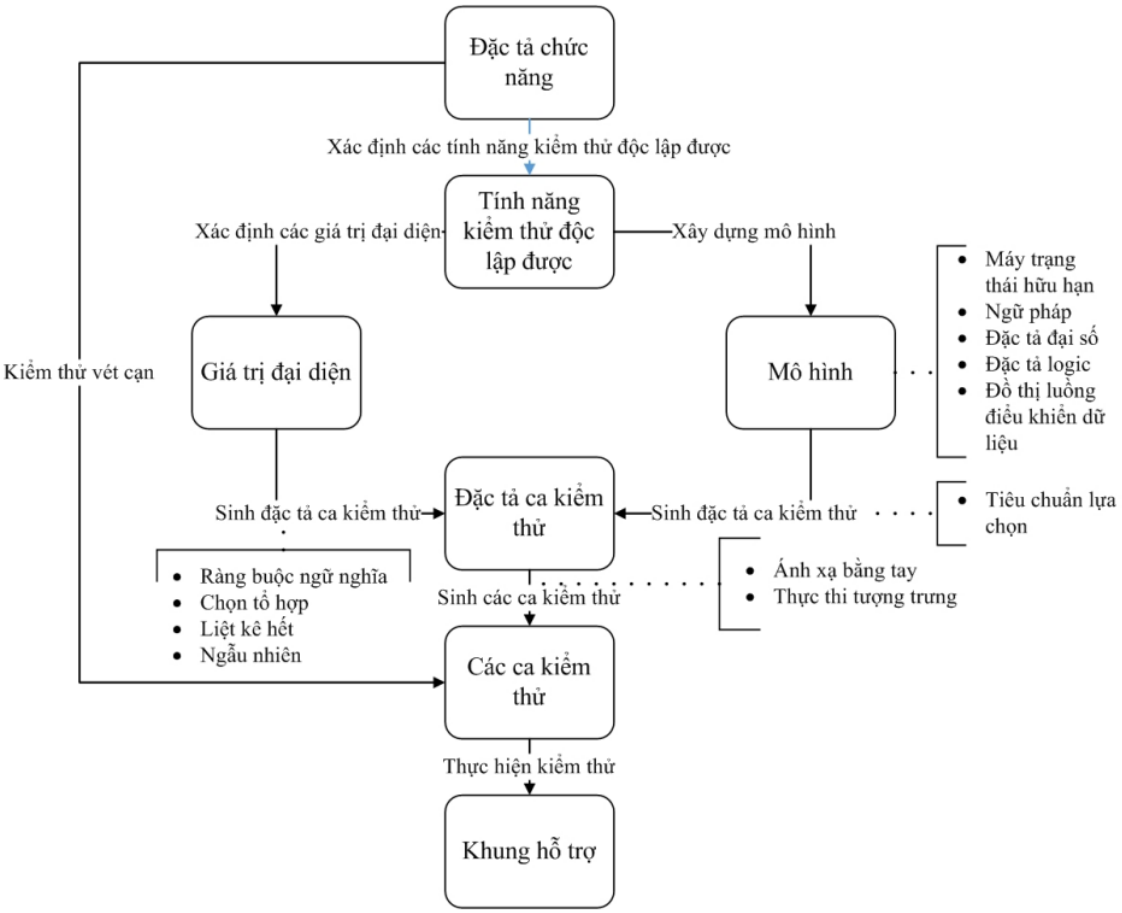
	Kiểm thử hộp đen	Kiểm thử hộp trắng
Khái niệm	-Là một phương pháp kiểm thử phần mềm mà việc kiểm tra các chức năng của một ứng dụng không cần quan tâm vào cấu trúc nội bộ hoặc hoạt động của nó.	Kiểm thử hộp trắng (While box test) là phương pháp thử nghiệm phần mềm, trong đó các thiết kế, cấu trúc giải thuật bên trong, và việc thực hiện các công việc đều được biết đến
Đối tượng kiểm thử	- Là thành phần phần mền (TPPM) có thể là 1 hàm chức năng, 1 modul chức năng, 1 phân hệ chức năng...	- Là 1 thành phần của phần mềm (1 chức năng, 1 module chức năng, 1 phân hệ chức năng....)
Phương pháp	-được áp dụng hầu như đến mọi cấp độ của kiểm thử phần mềm: +Kiểm thử đơn vị (Unit test) + Kiểm thử tích hợp (Intergration test) + Kiểm thử hệ thống (System test) + Kiểm thử chấp nhận (Acceptance test).	-Với những TPPM quá lớn sẽ tốn rất nhiều thời gian và công sức để kiểm thử nếu như dùng kiểm thử tích hợp (Integration test) hay kiểm thử chức năng (Functional test)). -Kỹ thuật white box test thích hợp dùng để kiểm thử đơn vị (Unit test)
Đặc điểm	- dựa vào thông tin duy nhất là các đặc tả về yêu cầu chức năng của TPPM tương ứng - Người kiểm thử không cần thiết phải có kiến thức về việc mã hoá, cấu trúc bên trong của TPPM, cũng như không yêu cầu phải biết lập trình phần mềm. - Việc kiểm thử được tiến hành dựa vào việc kiểm thử TPPM làm được gì, có phù hợp với yêu cầu của người dùng hay không. Các tester nhập số liệu vào phần mềm và chỉ cần xem kết quả của phần mềm và các mục tiêu kiểm tra. - Mức test này thường yêu cầu các tester phải viết test case đầy đủ trước khi test; khi test, đơn giản chỉ cần	-dựa vào giải thuật, cấu trúc bên trong chức năng của TPPM tương ứng. -Người kiểm thử phải có kiến thức nhất định về việc mã hoá, cấu trúc bên trong của chức năng, biết lập trình phần mềm. -Việc kiểm thử được tiến hành dựa vào việc kiểm xem giải thuật, mã lệnh đã làm có đúng không. -Mức test này thường yêu cầu các tester phải viết test case đầy đủ các nhánh trong code; khi test, sẽ set điều kiện và data để chạy vào đủ tất cả các nhánh trong giải thuật, đảm bảo thực hiện đầy đủ.

	thực hiện theo các bước mô tả trong test case thao tác và nhập data vào, sau đó xem kết quả trả về hoặc hành vi của phần mềm, rồi so sánh với kết quả mong đợi được viết trong testcase	
Tạo + thực hiện testcase	-Khi viết test case: Dựa vào yêu cầu và giao diện bên ngoài của chương trình (Không can thiệp vào bên trong code của chương trình) -Khi thực hiện test: Thực hiện trên giao diện của chương trình (yêu cầu chương trình phải chạy được mới test được, không can thiệp vào code)	- Khi viết test case: Dựa vào yêu cầu và nội dung Source Code (can thiệp vào bên trong Code của chương trình) - Khi thực hiện test: Thực thi test trong code (không cần thực thi chương trình, vì thực hiện test white box sẽ sử dụng framework nào đó hỗ trợ (Ví dụ như test kiểu debug)
VÍ DỤ:	Giả sử bạn có một ứng dụng thương mại điện tử. Trong kiểm thử hộp đen, bạn sẽ kiểm tra các chức năng như: Mua sắm, đăng sản phẩm, tạo tài khoản, hoặc hiệu suất làm việc của ứng dụng Bạn không cần phải biết ngôn ngữ lập trình hoặc làm thế nào các phần mềm đã được thực hiện. Các tester có thể được thực hiện bởi một cơ quan độc lập từ các developer, cho phép một cái nhìn khách quan và tránh sự phát triển thiên vị	

Câu 3: Thế nào là kiểm thử chức năng? Hãy nêu các bước chính của phương pháp kiểm thử chức năng bằng sơ đồ hình vẽ

Kiểm thử chức năng (functional testing) là các hoạt động kiểm tra chương trình dựa trên tài liệu mô tả chức năng, yêu cầu phần mềm, hay còn gọi là đặc tả chức năng (functional specification).

Đặc tả chức năng là tài liệu mô tả yêu cầu, hành vi mong muốn của chương trình. Tài liệu này là cơ sở để chúng ta tiến hành xây dựng các ca kiểm thử, thuật ngữ chuyên môn gọi là thiết kế kiểm thử (test design) và người thực hiện việc này là người thiết kế kiểm thử. Thiết kế kiểm thử là hoạt động chính trong kiểm thử chức năng.



Câu 4: Thế nào là kiểm thử giá trị biên? Cho một ví dụ theo ý hiểu của anh/ chị.

Kiểm thử giá trị biên (boundary value testing) là một trong những kỹ thuật được áp dụng phổ biến nhất trong cách tiếp cận kiểm thử chức năng (kiểm thử hộp đen).Là kỹ thuật thiết kế test case và hoàn thành phân vùng tương đương. Mục tiêu là lựa chọn các test case để thực thi giá trị biên vì lỗi thường xảy ra ở gần các giá trị biên này. Chương trình viết bằng ngôn ngữ không có kiểm tra kiểu mạnh càng cần kiểm thử giá trị biên Javascript, php, Visual Basic

- **VD: a <= x <=b thì sẽ chọn x= (a, a+1, a+b/2, b-1, b).**

Câu 5: Thế nào là kiểm thử lớp tương đương? Hãy áp dụng phương pháp kiểm thử lớp tương đương cho bài toán xét các trường hợp của tam giác?

Kiểm thử lớp tương đương là phương pháp chia miền dữ liệu kiểm thử thành các miền con sao cho dữ liệu trong mỗi miền con có cùng tính chất đối với chương trình, có nghĩa là các ca kiểm thử của một miền con sẽ cùng gây lỗi cho chương trình, hay cùng cho kết quả đúng, hay cùng cho kết quả sai tương tự nhau. Sau khi chia miền dữ liệu của chương trình thành các miền con tương đương, chúng ta chỉ cần chọn một phần tử đại diện của mỗi miền con này làm bộ dữ liệu kiểm thử. Các miền con này chính là các lớp tương đương.

Chúng ta áp dụng kiểm thử lớp tương đương cho bài toán tam giác, nhưng chúng ta áp dụng lớp tương đương cho đầu ra thay vì đầu vào. Ở đầu ra chúng ta thấy chương trình có thể in ra bốn khả năng: Không phải tam giác, Tam giác thường, Tam giác cân và Tam giác đều. Dựa trên các lớp đầu ra này, chúng ta có thể xác định bộ kiểm thử như sau

TT	a	b	c	Kết quả mong đợi
1	5	5	5	Tam giác đều
2	2	2	3	Tam giác cân
3	3	4	5	Tam giác không cân
4	4	1	2	Không là tam giác

Câu 6: Thế nào là kiểm thử bằng bảng quyết định? Nêu cấu trúc của một bảng quyết định.

Kiểm thử dựa trên bảng quyết định là phương pháp chính xác nhất trong các kỹ thuật kiểm thử chức năng. Bảng quyết định là phương pháp hiệu quả để mô tả các sự kiện, hành vi sẽ xảy ra khi một số điều kiện thỏa mãn.

Cấu trúc của một bảng quyết định chia thành bốn phần chính như sau:

+ Các biểu thức điều kiện C1, C2, C3;
+ Giá trị điều kiện T, F, –;
+ Các hành động A1, A2, A3, A4; và
+ Giá trị hành động, có (xảy ra) hay không. Chúng ta ký hiệu X để chỉ hành động là có xảy ra ứng với các điều kiện tương ứng của cột.

Câu 7: Hãy nêu các tính chất để áp dụng phương pháp kiểm thử bằng bảng quyết định cho bài toán.

Nên dùng kỹ thuật bảng quyết định cho các ứng dụng có một trong các tính chất sau:

– Chương trình có nhiều lệnh rẽ nhánh

– nhiều khối If-Then-Else

– Các biến đầu vào có quan hệ với nhau

– Có các tính toán giữa các biến đầu vào

– Có quan hệ nhân quả giữa đầu vào và đầu ra

– Có độ phức tạp chu trình (Cyclomatic) [McC76a] cao

Bảng quyết định không dễ áp dụng cho các bài toán lớn (với n điều kiện có 2n quy tắc).

Câu 8: Viết chương trình giải và biện luận phương trình bậc 2: $a.x^2 + b.x + c = 0$. Áp dụng kiểm thử giá trị biên để xây dựng bộ dữ liệu kiểm thử cho chương trình.

```
Program Giai_PT_Bac2
Uses crt;
Var a,b,c,d,x1,x2: real;
Begin
    Clrscr;
    Writeln('Giai phuong trinh bac 2');
    Writeln('-------------------------------------------------------');
    Writeln('Nhap he so a= '); readln(a);
    Writeln('Nhap he so b= '); readln(b);
    Writeln('Nhap he so c= '); readln(c);
    If a=0 then
       If b=0 then
          Writeln('Phuong trinh co vo so nghiem')
     Else writeln('Phuong trinh vo nghiem')
  Else writeln('Phuong trinh mot nghiem: x=', -c/b:4:2)
      Else
         Begin
            D := b*b-4*a*c;
           If D=0 then writeln('Phuong trinh co nghiem kep: x=', -b/(2*a):4:2)
```

```
        Else
            If D<0 then writeln('Phuong trinh vo nghiem')
            Else
             Begin
               x1 := (-b-sqrt(D))/(2*a);
               x2 := (-b+sqrt(D))/(2*a);
               writeln('Phuong trinh co 2 nghiem la x1= ', x1:4:2, 'x2=', x2:4:2);
            End;
    End;
Readln;
End.
```

Kiểm thử giá trị biên

Phương trình $ax^2+bx+c=0$ có miền xác định là □ . Để nó là phương trình bậc hai thì *a* phải khác 0. Giả sử các giá trị *a*, *b*, *c* nằm trong đoạn $[-5,5]$. Ta có bảng kiểm thử các giá trị biên cho bài toán giải phương trình bậc hai như sau:

Case	*a*	*b*	*c*	Expected Output
1	5	5	5	Vô nghiệm
2	5	5	4	Vô nghiệm
3	5	4	5	Vô nghiệm
4	4	5	5	Vô nghiệm
5	5	4	4	Vô nghiệm
6	4	5	4	Vô nghiệm
7	4	4	5	Vô nghiệm
8	5	5	-5	Có 2 nghiệm $x_1 = 0.618$ $x_2 = -1.618$
9	5	-5	5	Vô nghiệm

10	-5	5	5	Có 2 nghiệm $x_1 = 1.618$ $x_2 = -0.618$
11	5	-5	-5	Có 2 nghiệm $x_1 = 1.618$ $x_2 = -0.618$
12	-5	5	-5	Vô nghiệm
13	-5	-5	5	Có 2 nghiệm $x_1 = 0.618$ $x_2 = -1.618$
14	4	5	-5	Có 2 nghiệm $x_1 = 0.6559$ $x_2 = -1.9059$
15	5	4	-5	Có 2 nghiệm $x_1 = 0.677$ $x_2 = -1.477$
16	5	-5	4	Vô nghiệm
17	-5	5	4	Có 2 nghiệm $x_1 = 1.5247$ $x_2 = -0.5247$
18	4	-5	5	Vô nghiệm
19	4	-4	5	Vô nghiệm
20	4	5	-4	Có 2 nghiệm $x_1 = 0.5542$ $x_2 = -1.8042$
21	4	4	-5	Có 2 nghiệm $x_1 = 0.7247$ $x_2 = -1.7247$

Câu 9: Viết chương trình giải và biện luận phương trình bậc 2: $a.x^2 + b.x + c = 0$. Áp dụng kiểm thử lớp tương đương để xây dựng bộ dữ liệu kiểm thử cho chương trình.

```
Program Giai_PT_Bac2
Uses crt;
Var a,b,c,d,x1,x2: real;
Begin
    Clrscr;
    Writeln('Giai phuong trinh bac 2');
    Writeln('-----------------------------------------------------');
    Writeln('Nhap he so a= '); readln(a);
    Writeln('Nhap he so b= '); readln(b);
    Writeln('Nhap he so c= '); readln(c);
    If a=0 then
       If b=0 then
          Writeln('Phuong trinh co vo so nghiem')
     Else writeln('Phuong trinh vo nghiem')
  Else writeln('Phuong trinh mot nghiem: x=', -c/b:4:2)
      Else
         Begin
            D := b*b-4*a*c;
           If D=0 then writeln('Phuong trinh co nghiem kep: x=', -b/(2*a):4:2)
```

```
        Else
            If D<0 then writeln('Phuong trinh vo nghiem')
            Else
             Begin
               x1 := (-b-sqrt(D))/(2*a);
               x2 := (-b+sqrt(D))/(2*a);
               writeln('Phuong trinh co 2 nghiem la x1= ', x1:4:2, 'x2=', x2:4:2);
            End;
    End;
Readln;
End.
```

Kiểm thử tương đương

- Miền đầu ra cho các giá trị:
  - Không phải là phương trình bậc hai.
  - Phương trình vô nghiệm.
  - Phương trình có nghiệm kép.
  - Phương trình có hai nghiệm phân biệt.
- Xác định các lớp tương đương:
  - $A_1 = \{< a,b,c > | \ a = 0\}$.
  - $A_2 = \{< a,b,c > | \ a \neq 0 \& b^2 - 4ac < 0\}$
  - $A_3 = \{< a,b,c > | \ a \neq 0 \& b^2 - 4ac = 0\}$
  - $A_4 = \{< a,b,c > | \ a \neq 0 \& b^2 - 4ac > 0\}$
- Bảng kiểm thử

Case	*a*	*b*	*c*	Expected Output
1	0	0	9	Không phải là

				phương trình bậc hai
2	1	1	1	Phương trình vô nghiệm
3	1	-2	1	Phương trình có nghiệm kép $x=1$
4	1	-3	2	Phương trình có hai nghiệm phân biệt: $x_1=2$ $x_2=1$

Câu 10: Viết chương trình giải và biện luận phương trình bậc 2: $a.x^2 + b.x + c = 0$. Áp dụng kiểm thử bằng bảng quyết định để xây dựng bộ dữ liệu kiểm thử cho chương trình.

1. **Xây dựng bảng quyết định cho bài toán giải phương trình bậc hai.**

Điều kiện	1	2	3	4
C1: $a \neq 0$?	T	F	F	F
C2: $b^2-4ac<0$?	-	T	F	F
C3: $b^2-4ac=0$?	-	-	T	F
Số luật	4	2	1	1
A1: Không phải phương trình bậc hai	X			
A2: Phương trình vô nghiệm		X		
A3: Phương trình có một nghiệm kép			X	
A4: Phương trình có hai nghiệm phân biệt				X

Câu 11: Xây dựng bảng quyết định cho hàm Triangle. Kiểm tra các trường hợp của tam giác.

- Phân tích bài toán: Giả sử a, b, c là độ dài ba cạnh của tam giác. Khi đó:
  - Các biểu thức điều kiện:
    - a < b + c
    - b < a + c
    - c < a + b
    - a = b
    - a = c
    - b = c
  - Các hành động:
    - Không là tam giác
    - Tam giác thường
    - Tam giác cân
    - Tam giác đều
    - Không hợp lệ
- Bảng quyết định cho bài toán Triangle

Điều kiện	1	2	3	4	5	6	7	8	9	10	11
C1: a<b+c?	F	T	T	T	T	T	T	T	T	T	T
C2: b<a+c?	-	F	T	T	T	T	T	T	T	T	T
C3: c<a+b?	-	-	F	T	T	T	T	T	T	T	T
C4: a=b?	-	-	-	T	T	T	T	F	F	F	F
C5: a=c?	-	-	-	T	T	F	F	T	T	F	F
C6: b=c?	-	-	-	T	F	T	F	T	F	T	F
Số luật	32	16	8	1	1	1	1	1	1	1	1
A1: Không là tam giác	X	X	X								
A2: Tam giác thường											X
A3: Tam giác cân							X		X	X	
A4: Tam giác đều				X							
A5: Không hợp lệ					X	X		X			

- Nếu các điều kiện chỉ là T và F ta có 2^n cột quy tắc. Mỗi một giá trị "-" sẽ đại diện cho hai cột. Tổng số luật bằng 2^n tức là số cột quy tắc đã đủ.
- Sử dụng bảng quyết định cho Triangle ta có 11 ca kiểm thử:
  - 3 trường hợp không là tam giác
  - 1 trường hợp tam giác thường
  - 3 trường hợp là tam giác cân
  - 1 trường hợp tam giác đều
  - 3 trường hợp không hợp lệ
- Các ca kiểm thử bằng bảng quyết định cho Triangle

STT	A	B	C	Kết quả mong đợi
1	4	1	2	Không là tam giác
2	1	4	2	Không là tam giác
3	1	2	4	Không là tam giác
4	5	5	5	Tam giác đều
5	?	?	?	Không hợp lệ
6	?	?	?	Không hợp lệ
7	2	2	3	Tam giác cân
8	?	?	?	Không hợp lệ
9	2	3	2	Tam giác cân
10	3	2	2	Tam giác cân
11	3	4	5	Tam giác thường

Câu 12: Xây dựng hàm kiểm tra dữ liệu hợp lệ của hồ sơ học sinh gồm các trường Họ Tên, Ngày Sinh, Email và Giới tính (Nam, Nữ). Hãy cài đặt hàm này và áp dụng kiểm thử lớp tương đương để viết các kiểm thử đơn vị bằng ngôn ngữ C.

```
typedef struct hocsinh {

char    hoten[35];

char ngaysinh [30];

char email [60];

char gioitinh [10];

} HOCSINH;

HOCSINH danhsach [MAX];

int numrecords = 0;

void nhapmoi()

{

  Int done = 0;
```

```
  char hoten[35];
  char ngaysinh[30];
  char email [50];
  char gioitinh [10];
  do{
            printf("\nHo ten: ");
            gets(hoten);
             if (strlen(hoten) ==0)
                done = 1;
           else
              {
strcpy (danhsach[numrecord].hoten, hoten);
            printf("\nNgay sinh: ");
            gets(danhsach[numrecords].ngaysinh);
            printf("\nEmail: ");
 danhsach[numrecord].email;
        printf("\nGioi tinh: ");
danhsach[numrecords].gioitinh;
        numrecords++;
                    }
```

```
        } while (!done);

    }
```

Câu 14: cho đoạn lệnh

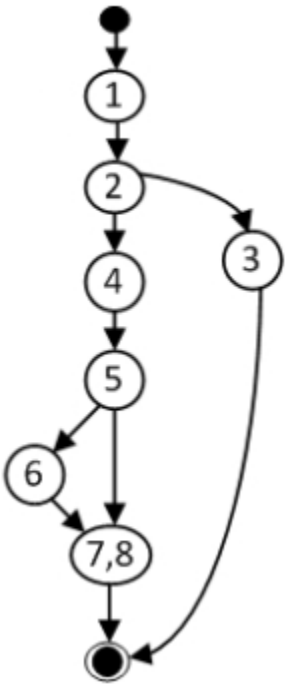
Float mu(int a, int b, int c)

{

1. **Float t;**
2. **If (a = = 0)**
3. **Return 0;**
4. **Int y= 0;**
5. **If((a==b)||(c==d))**
6. **y =1;**
7. **t = 1/y;**
8. **Return t;**

}

xây dựng đồ thị dòng điều kiển của nó



Câu 15: Cho đoạn lệnh sau:

```
int  giaithua( int n)
{
    1. int  i, t=1;
    2. if  (n = =0)
    3.      t =1;
    4. Else if(n > 0)
    5. {
    6. While (i<=n)
    7.     {
    8.        t = t * i;
    9.        i =  i + 1;
    10.}
    11.Return t;
}
```

xây dựng đồ thị dòng điều kiển của thuật toán trên

Câu 16: Hãy nêu các độ đo kiểm thử.

a) Uni Test (Kiểm tra mức đơn vị):

- Là hoạt động kiểm thử thực hiện trên các hàm, lớp,... hay thành phần riêng lẻ.
- Cần hiểu biết về thiết kế chương trình và code, thực hiện bởi Lập trình viên
- Mục đích: cô lập từng thành phần của chương trình và chứng minh các bộ phận riêng lẻ chính xác về các yêu cầu chức năng

a) Intergration Test (Kiểm tra tích hợp)

- Nhằm phát hiện lỗi giao tiếp xảy ra giữa các thành phần cũng như lỗi của bản thân từng thành phần (nếu có).
- Integration Test có 2 mục tiêu chính:
  + Phát hiện lỗi giao tiếp xảy ra giữa các Unit
  + Tích hợp các Unit đơn lẻ thành các hệ thống nhỏ (subsystem) và cuối cùng là nguyên hệ thống hoàn chỉnh (system) chuẩn bị cho kiểm tra ở mức hệ thống (System Test).
- Có 4 loại kiểm tra trong Integration Test.
  + Kiểm tra cấu trúc (structure): Tương tự White Box Test.
  + Kiểm tra chức năng (functional): Tương tự Black Box Test
  + Kiểm tra hiệu năng (performance): Kiểm tra việc vận hành của hệ thống.
  + Kiểm tra khả năng chịu tải (stress): Kiểm tra các giới hạn của hệ thống.

b) System Test (Kiểm tra mức hệ thống)

- Là một mức của tiến trình kiểm thử phần mềm khi các module và tích hợp các module đã được test.
- Mục tiêu: đánh giá phần mềm có tuân thủ theo các yêu cầu đã đưa ra không
- Điểm khác nhau then chốt giữa **Integration Test** và **System Test** là System Test chú trọng các hành vi và lỗi trên toàn hệ thống, còn Integration Test chú trọng sự giao tiếp giữa các đơn thể hoặc đối tượng khi chúng làm việc cùng nhau

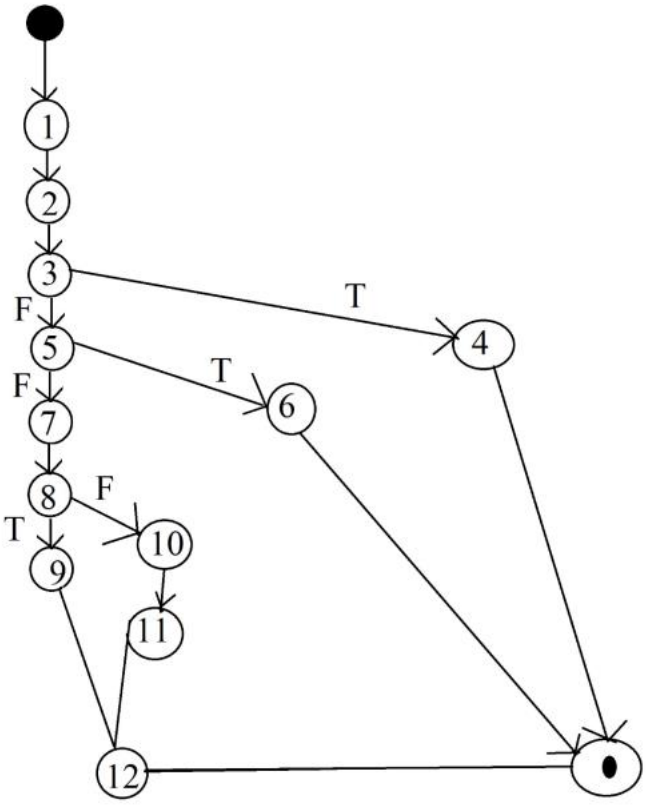
c) Acceptance test (kiểm tra chấp nhận):

- Kiểm thử chấp nhận là một cấp độ trong tiến trình kiểm thử phần mềm nhằm kiểm thử hệ thống về khả năng chấp nhận được
- Mục tiêu của kiểm thử này là để đánh giá sự tuân thủ của hệ thống với các yêu cầu nghiệp vụ và thẩm định xem đã có thể chấp nhận để bàn giao chưa
- Gồm 2 loại kiểm thử là:

- Alpha Test, người dùng kiểm thử phần mềm ngay tại nơi phát triển phần mềm, lập trình viên sẽ ghi nhận các lỗi hoặc phản hồi, và lên kế hoạch sửa chữa.
- Beta Test, phần mềm sẽ được gửi tới cho người dùng để kiểm thử ngay trong môi trường thực, lỗi hoặc phản hồi cũng sẽ gửi ngược lại cho lập trình viên để sửa chữa.

Câu 17: Áp dụng kiểm thử cho độ đo C_1 cho đoạn lệnh của hàm UCLN

```
int     UCLN(int m, int n)
{
1.  if (m < 0) m = -m;
2.  if (n < 0) n = -n;
3.  if (m == 0)
4.      return n;
5.  if (n == 0)
6.     return m;
7.  while (m != n)
    {
8.  if(m > n)
9.        m = m – n;
10.   else
11.       n = n – m;
    }
1.   return m;
     }
```


1
2
3
T
F
4
5
T
6
F
7
8
F
T
10
9
11
12

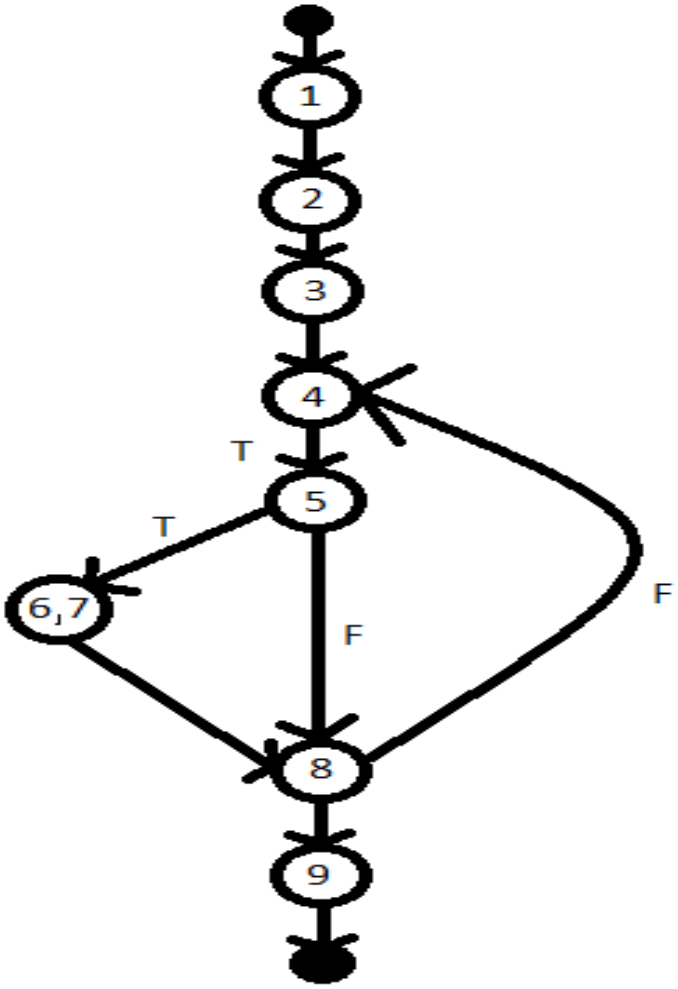
Câu 18: Áp dụng kiểm thử cho độ đo C_2 cho đoạn lệnh của hàm UCLN

Câu 19: Áp dụng kiểm thử cho độ đo C_3 cho đoạn lệnh của hàm UCLN

Câu 20: cho đoạn lệnh sau:

```
Float  Trungbinhcong( int a[10], int n)
{
1.  int i, tong=0, dem=0;
2.  float tb;
3.  i =1;
4.  While( i<=n)
5.  If   (a[i] %2 = =0)
                {
6.                  Tong = tong + a[i];
7.                  Dem = dem +1;
                }
8.tTb = (float) tong/dem;
9. return tb;
}
```

Xây dựng đồ thị dòng điều khiển ứng với độ đo C_2



Có 2 đường đi

1): 1;2;3;4;5(T);6;7;8;9

2):1;2;3;4;5(F);8,9

Câu 21: phân biệt kiểm thử hộp trắng và kiểm thử họp đen

	Kiểm thử hộp đen	Kiểm thử hộp trắng
Khái niệm	-Là một phương pháp kiểm thử phần mềm mà việc kiểm tra các chức năng của một ứng dụng không cần quan tâm vào cấu trúc nội bộ hoặc hoạt động của nó.	Kiểm thử hộp trắng (While box test) là phương pháp thử nghiệm phần mềm, trong đó các thiết kế, cấu trúc giải thuật bên trong, và việc thực hiện các công việc đều được biết đến
Đối tượng kiểm thử	- Là thành phần phần mền (TPPM) có thể là 1 hàm chức năng, 1 modul chức năng, 1 phân hệ chức năng...	- Là 1 thành phần của phần mềm (1 chức năng, 1 module chức năng, 1 phân hệ chức năng....)
Phương pháp	-được áp dụng hầu như đến mọi cấp độ của kiểm thử phần mềm: +Kiểm thử đơn vị (Unit test) + Kiểm thử tích hợp (Intergration test) + Kiểm thử hệ thống (System test) + Kiểm thử chấp nhận (Acceptance test).	-Với những TPPM quá lớn sẽ tốn rất nhiều thời gian và công sức để kiểm thử nếu như dùng kiểm thử tích hợp (Integration test) hay kiểm thử chức năng (Functional test)). -Kỹ thuật white box test thích hợp dùng để kiểm thử đơn vị (Unit test)
Đặc điểm	- dựa vào thông tin duy nhất là các đặc tả về yêu cầu chức năng của TPPM tương ứng - Người kiểm thử không cần	-dựa vào giải thuật, cấu trúc bên trong chức năng của TPPM tương ứng. -Người kiểm thử phải có kiến

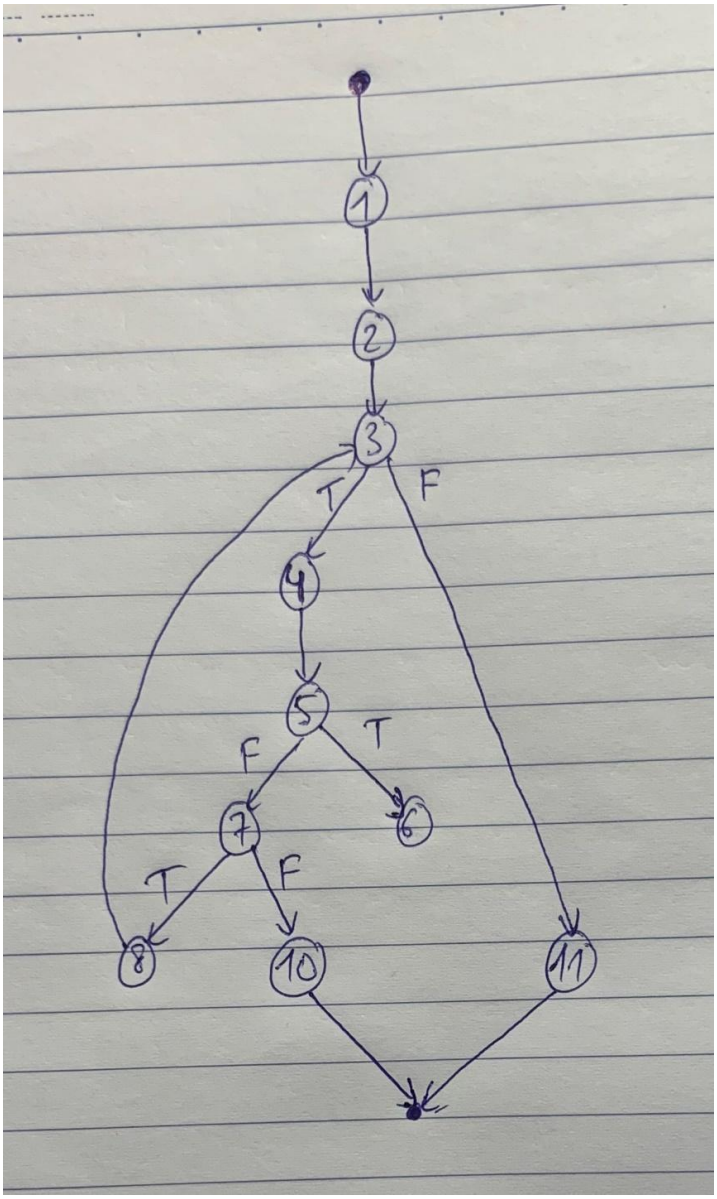
	thiết phải có kiến thức về việc mã hoá, cấu trúc bên trong của TPPM, cũng như không yêu cầu phải biết lập trình phần mềm. - Việc kiểm thử được tiến hành dựa vào việc kiểm thử TPPM làm được gì, có phù hợp với yêu cầu của người dùng hay không. Các tester nhập số liệu vào phần mềm và chỉ cần xem kết quả của phần mềm và các mục tiêu kiểm tra. - Mức test này thường yêu cầu các tester phải viết test case đầy đủ trước khi test; khi test, đơn giản chỉ cần thực hiện theo các bước mô tả trong test case thao tác và nhập data vào, sau đó xem kết quả trả về hoặc hành vi của phần mềm, rồi so sánh với kết quả mong đợi được viết trong testcase	thức nhất định về việc mã hoá, cấu trúc bên trong của chức năng, biết lập trình phần mềm. -Việc kiểm thử được tiến hành dựa vào việc kiểm xem giải thuật, mã lệnh đã làm có đúng không. -Mức test này thường yêu cầu các tester phải viết test case đầy đủ các nhánh trong code; khi test, sẽ set điều kiện và data để chạy vào đủ tất cả các nhánh trong giải thuật, đảm bảo thực hiện đầy đủ.
Tạo + thực hiện testcase	-Khi viết test case: Dựa vào yêu cầu và giao diện bên ngoài của chương trình (Không can thiệp vào bên trong code của chương	- Khi viết test case: Dựa vào yêu cầu và nội dung Source Code (can thiệp vào bên trong

	trình) -Khi thực hiện test: Thực hiện trên giao diện của chương trình (yêu cầu chương trình phải chạy được mới test được, không can thiệp vào code)	Code của chương trình) - Khi thực hiện test: Thực thi test trong code (không cần thực thi chương trình, vì thực hiện test white box sẽ sử dụng framework nào đó hỗ trợ (Ví dụ như test kiểu debug)

Câu 22: Cho hàm được viết bằng ngôn ngữ C hàm BinSearch

```
Int  Binsearch(int x, int v[], int n)
{
    1.  int low = 0, high, mid;
    2.   high = n - 1;
    3.  while (low <= high)
    {
    4.   mid = (low + high)/2;
    5.  if (x < v[mid])
    6.  high = mid - 1;
    7.  else if (x > v[mid])
    8.          low = mid + 1;
    9.  else
    10.return mid;
    }
    11.return -1;
 }
```

Xây dựng đồ thị dòng điều khiển cho hàm BinSearch ứng với độ đo C_1



Bảng kiểm thử:

Test path	input	EO	RO	note
1	1,2,3(T),4,5(T),6,11	4,5,50	1	
2	1,2,3(F),11	1,2,-10	0	

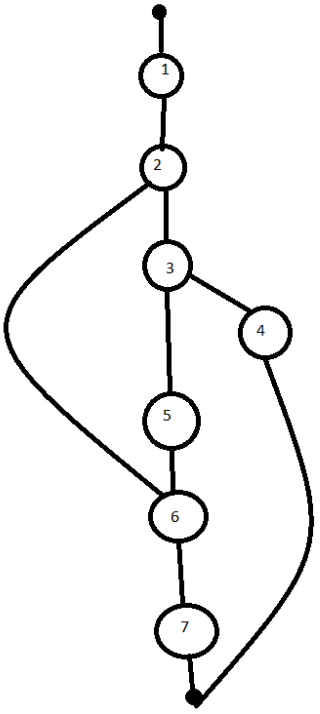
Câu 25. Cho đoạn lệnh kiểm tra số nguyên tố được viết bằng ngôn ngữ C

int LaSoNguyenTo(int n)

```
{ int i=2;
Do
{
        if((n % i) == 0)
        return 0;
        i++;
} while(i <= n/2);
        return 1;
}
```

Xây dựng đồ thị dòng điều khiển cho hàm LaSoNguyenTo ứng với độ đo C2.

```
int LaSoNguyenTo(int n){
        1. int i=2;
        2. Do{
        3. if((n % i) == 0)
        4. return 0;
        5. i++;
         } 6.while(i <= n/2);
        7. return 1;
}
```



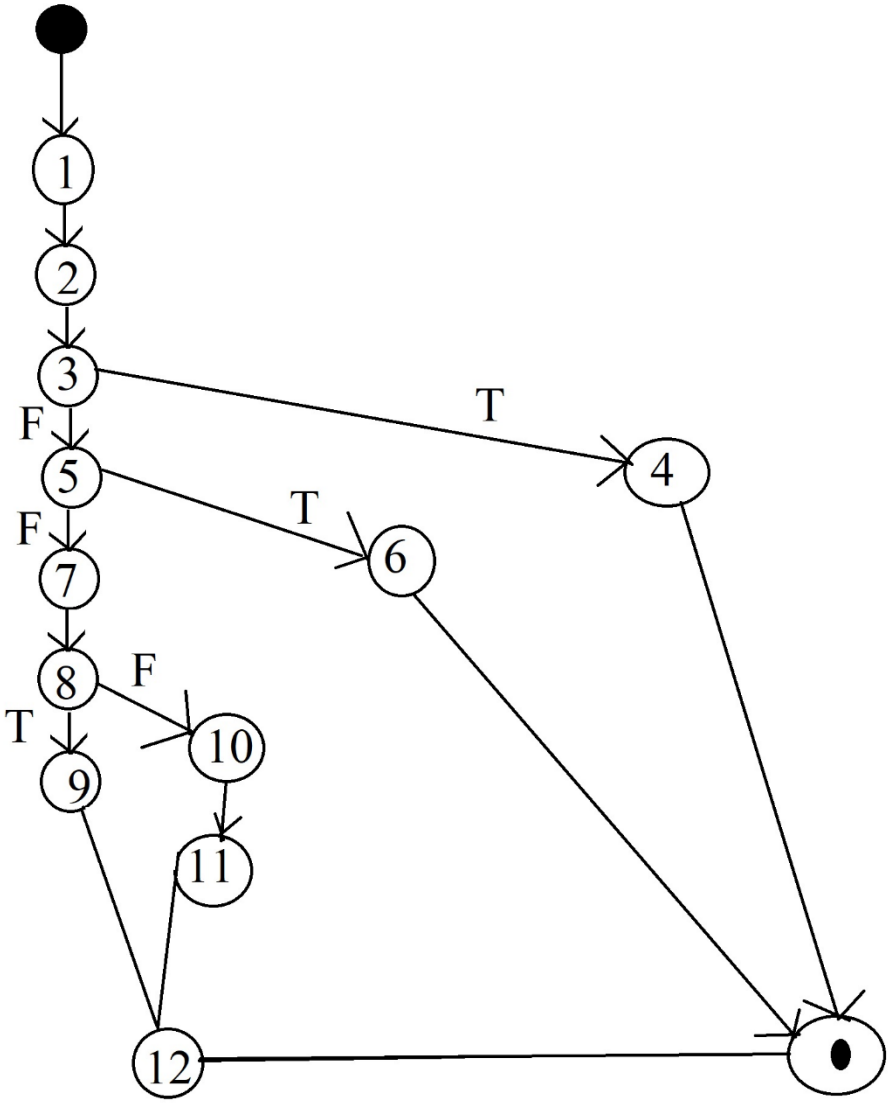
Test case	Input	output	Real output
1	3	1	1
2	7	1	1
3	15	0	0
4	17	1	1
5	20	0	0
6	2	1	0

Bảng kiểm thử độ đo C2

Câu 29: Cho hàm tìm UCLN được viết bằng ngôn ngữ C

```
int     UCLN(int m, int n)
{
        9.  if (m < 0) m = -m;
        10.if (n < 0) n = -n;
        11.if (m == 0)
        12.    return n;
        13.if (n == 0)
        14.  return m;
        15.while (m != n)
        {
        16.if(m > n)
        9.          m = m – n;
        10.    else
         11.        n = n – m;
        }
     2.   return m;
         }
```

• Hãy xây dựng đồ thị dòng điều khiển cho hàm UCLN ứng với độ đo C1


1
2
3
T
F
4
5
T
F
6
7
8
F
T
10
9
11
12