HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA/VIỆN/BỘ MÔN: CÔNG NGHỆ THÔNG TIN/CÔNG NGHỆ PHẦN MỀM

NGÂN HÀNG CÂU HỎI THI TỰ LUẬN

Tên học phần: Đảm bảo chất lượng phần mềm **Mã học phần: INT 416**
Ngành đào tạo: **Trình độ đào tạo:**

1. Ngân hàng câu hỏi thi

- **Câu hỏi loại 1 điểm**

Câu hỏi 1.1: Lỗi phần mềm là gì? Nguyên nhân gây ra lỗi phần mềm?
Câu hỏi 1.2: Cơ sở để kiểm định chất lượng phần mềm?
Câu hỏi 1.3: Đảm bảo phần mềm xuất phát từ đâu? Tiến triển của nó như thế nào?
Câu hỏi 1.4: Kể ra các độ đo đặc trưng chất lượng chính của McCall? Giải thích nội dung của nó?
Câu hỏi 1.5: Nêu các đặc trưng chất lượng theo Hawlett? Giải thích nội dung mỗi loại?
Câu hỏi 1.6: Trình bày kỹ thuật Walkthrough
Câu hỏi 1.7: Trình bày kỹ thuật Inspection
Câu hỏi 1.8: Trình bày tóm tắt SQA trong tiêu chuẩn ISO 9000-3
Câu hỏi 1.9: Trình bày các đặc tính chất lượng ISO 9126.

Câu hỏi 1.10: Trình bày tóm tắt SQA trong tiêu chuẩn IEEE std1028
Câu hỏi 1.11: Trình bày các mức tiêu chuẩn trong CMM?
Câu hỏi 1.12: Mục tiêu của SQA là gì? Các hoạt động chính đảm bảo chất lượng phần mềm là những hoạt động nào?
Câu hỏi 1.13: Khảo sát nhu cầu SQA gồm những nội dung gì? Nhằm trả lời các câu hỏi gì?
Câu hỏi 1.14: Trình bày cấu trúc tổ chức đơn vị SQA?
Câu hỏi 1.15: Ca kiểm thử là cái gì? Mục tiêu thiết kế ca kiểm thử?
Câu hỏi 1.16: Kiểm thử hộp trắng là gì? Nêu các đặc trưng của nó?
Câu hỏi 1.17: Kiểm thử hộp đen là gì? Nêu các đặc trưng của nó?
Câu hỏi 1.18: Có những loại công cụ tự động nào trợ giúp kiểm thử, mô tả nội dung của mỗi loại?
Câu hỏi 1.19: Ai là người phải tham gia kiểm thử phần mềm? Nêu vai trò và trách nhiệm của mỗi đối tượng?
Câu hỏi 1.20: Đồ thị luồng điều khiển gồm những yếu tố nào? Đồ thị luồng điều khiển dùng để làm gì?
Câu hỏi 1.20: Đồ thị luồng dữ liệu gồm những yếu tố nào? Đồ thị luồng dữ liệu dùng để làm gì?
Câu hỏi 1.21: Nêu các loại điều kiện trong cấu trúc điều khiển và cho ví dụ? Có những loại sai nào trong điều kiện khi kiểm thử?
Câu hỏi 1.22: Chiến lược kiểm thử phân nhánh nghĩa là gì? Yêu cầu đặt ra cho kiểm thử phân nhánh là gì?

Câu hỏi 1.23: Kiểm thử đơn vị là gì? Hoạt động kiểm thử đơn vị gồm những nội dung gì? Nó liên quan đến những nhân tố nào?
Câu hỏi 1.24: Kiểm thử tích hợp là gì? Kiểm thử tích hợp thực hiện khi nào?
Câu hỏi 1.25: Nội dung chính của kiểm thử hệ thống? Nêu một số câu hỏi đặt ra cho kiểm thử hệ thống?
Câu hỏi 1.26: Khi nào nên dùng test tools? Ưu/nhược của việc dùng Test tools?
Câu hỏi 1.27: Kể tên một vài test tools cho kiểm thử chức năng, hiệu năng?
Câu hỏi 1.28: Quy trình kiểm thử phần mềm nói chung?
Câu hỏi 1.29: Test plan là gì, gồm những nội dung gì?

- **Câu hỏi loại 2 điểm**

Câu hỏi 2.1: Tại sao phải kiểm thử phần mềm? Mục tiêu kiểm thử là gì?
Câu hỏi 2.2: Giải thích sự khác nhau giữa *validation và verification.*
Câu hỏi 2.3: Giải thích sự khác nhau giữa *failure, error, và fault.*
Câu hỏi 2.4: Điểm mạnh và điểm yếu của kiểm thử tự động và kiểm thử bằng tay?
Câu hỏi 2.5: Kiểm thử Beta là cái gì? Kiểm thử Alpha là cái gì? Nêu sự giống và khác nhau cơ bản giữa chúng?
Câu hỏi 2.6: Nêu các bước kiểm thử tích hợp từ trên xuống? Ưu nhược điểm của cách tiếp cận này?
Câu hỏi 2.7: Nêu các bước kiểm thử tích hợp từ dưới lên? Ưu nhược điểm của cách tiếp cận này?
Câu hỏi 2.8: Thế nào là một ca kiểm thử tốt? ca kiểm thử thành công? Lợi ích phụ của kiểm thử là gì?
Câu hỏi 2.9: Giải thích sự khác nhau giữa kiểm thử hộp trắng và kiểm thử hộp đen?
Câu hỏi 2.10: Nếu chương trình thành công ở tất cả các bộ kiểm thử của kiểm thử hộp đen. Liệu ta có cần kiểm thử hộp trắng nữa không? Tại sao?

Câu hỏi 2.11: Khi nào dùng kỹ thuật bảng quyết định, kiểm thử biên, kiểm thử theo cặp?

Câu hỏi 2.12: Khi nào dùng kỹ thuật kiểm thử biên, kiểm thử theo cặp, sơ đồ chuyển trạng?
Câu hỏi 2.13: các thành phần cần có trong test plan?
Câu hỏi 2.14: Các giai đoạn chính của 1 tiến trình test?

Bài tập
Câu hỏi 2.15: Cho Form gồm 1 ratio button Nữ, 1 ratio button Độc thân và 1 Danh sách chọn Chuyên ngành gồm: CNTT, QTKD, VT, Kế toán.
a. Nếu kiểm thử tất cả các trường hợp xảy ra thì cần bao nhiêu test cases,
b. Mỗi test case chứa bao nhiêu cặp giá trị?
c. Liệt kê các cặp giá trị có thể xảy ra?
d. Thiết kế bộ pairwise test suite (kiểm thử theo cặp)

Câu hỏi 2.16: Một phần mềm điều khiển chơi game đơn giản giữa 2 người chơi. Mỗi người điều khiển một nút bấm để đưa bóng vào lỗ. Mỗi lần bóng vào lỗ, người đó được cộng 1 điểm. Ai được 5 điểm trước sẽ thắng. Nếu bóng không vào lỗ, người còn lại được quyền điều khiển bóng.
a. Lập sơ đồ chuyển trạng thái

b. Xác định các đường chạy để phủ hết các cạnh

c. Thiết kế test case tương ứng

Câu hỏi 2.17: Thông tin về block1 bao gồm size(small, large), color(red, green, blue), shape (circle, triangle, square).

a. Nếu kiểm thử tất cả các trường hợp xảy ra thì cần bao nhiêu ca kiểm thử?

b. Số cặp tối đa mà một ca kiểm thử có thể chứa

c. Xác định các cặp giá trị có thể xảy ra

d. Thiết kế bộ kiểm thử theo cặp (pairwise test suite)

di.

Câu hỏi 2.18: Cần phát triển module kiểm tra điều kiện dự thi của sinh viên gồm: Nếu sinh viên đi học >=80% số buổi, điểm giữa kì >0, điểm bài tập lớn >0 sẽ được thi. Nếu sinh viên đủ điều kiện dự thi và có điểm bài tập lớn = 10 hoặc điểm giữa kỳ = 10 sẽ được miễn thi.

a. Dùng kỹ thuật bảng quyết định để xác định test cases

b. Dùng kỹ thuật đồ thị nguyên nhân kết quả để xác định test cases

Câu hỏi 2.19: Cho hệ thống S nhận n tham số đầu vào, mỗi tham số có m giá trị. Trả lời câu hỏi:

a. Số cặp tối đa mà một ca kiểm thử chứa tối đa bao nhiêu cặp

b. Trong trường hợp lý tưởng, ta cần bao nhiêu ca kiểm thử để bao phủ tất cả các cặp của hệ thống?

c. Tính tổng số cặp mà bộ kiểm thử phải bao phủ?

d. Cho n = 13, m = 3. Số ca kiểm thử tối thiểu cần chọn để thu được bộ kiểm thử theo cặp (pairwise test suite)?

Câu hỏi 2.20: Form đăng ký mua vé tàu được cho như hình vẽ. Danh sách ga ở Ga đi và Ga đến là {Hà Nội, Vinh, Huế, Đà Nẵng, Sài Gòn}. Danh sách mác tàu là {SE,TN}. Không tính

trường Ngày đi, hãy thực hiện:

a. Nếu kiểm thử tất cả các trường hợp xảy ra thì cần bao nhiêu ca kiểm thử?

b. Số cặp tối đa mà một ca kiểm thử có thể chứa

c. Xác định các cặp giá trị có thể xảy ra

d. Thiết kế bộ kiểm thử theo cặp (pairwise test suite)

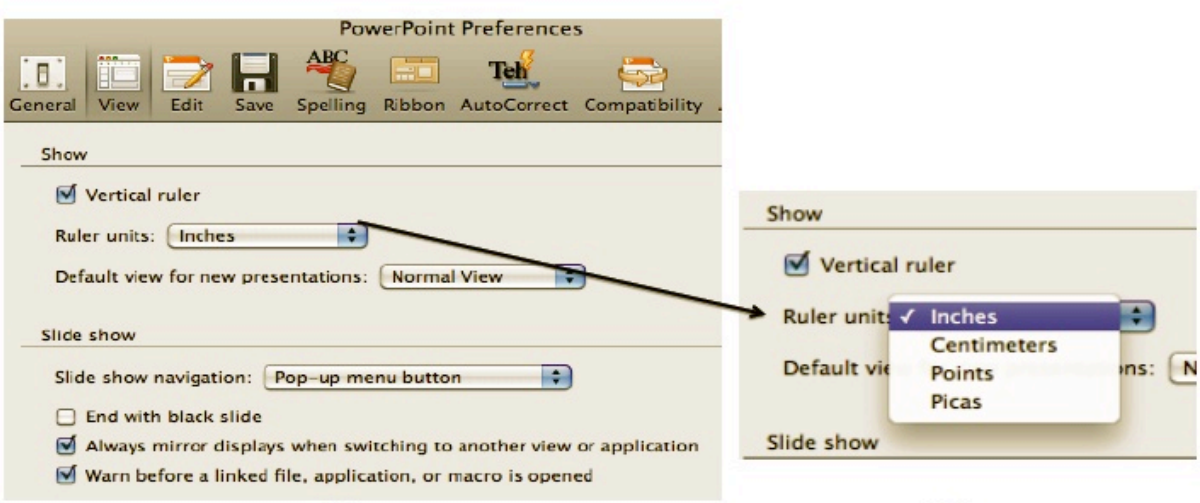
Câu hỏi 2.21: Xét tab tùy chọn View từ một phiên bản của phần mềm Powerpoint Microsoft. Bảng (c) chính là dữ liệu sau trích xuất.

a. Nếu kiểm thử tất cả các trường hợp xảy ra thì cần bao nhiêu ca kiểm thử?

b. Số cặp tối đa mà một ca kiểm thử có thể chứa

c. Xác định các cặp giá trị có thể xảy ra

d. Thiết kế bộ kiểm thử theo cặp (pairwises test suite)



(a) (b)

Vertical Ruler	Ruler Units	Default View	SS Navigation	End with Black	Always Mirror	Warn Before
Visible	Inches	Normal	Pop-up	Yes	Yes	Yes
Invisible	Centimeters	Slide	None	No	No	No
	Points	Outline				
	Picas					

(c)

Câu hỏi 2.22: Cần phát triển module tính thuế thu nhập cá nhân dựa trên phần thu nhập tính thuế. Biểu thuế thu nhập cá nhân được cho như bảng dưới:

Bậc thuế	Phần thu nhập tính thuế/năm (triệu đồng)	Phần thu nhập tính thuế/tháng (triệu đồng)	Thuế suất (%)
1	Đến 60	Đến 5	5
2	Trên 60 đến 120	Trên 5 đến 10	10
3	Trên 120 đến 216	Trên 10 đến 18	15
4	Trên 216 đến 384	Trên 18 đến 32	20
5	Trên 384 đến 624	Trên 32 đến 52	25
6	Trên 624 đến 960	Trên 52 đến 80	30
7	Trên 960	Trên 80	35

(a) Thiết kế các ca kiểm thử dùng kỹ thuật phân tích giá trị biên
(b) Thiết kế các ca kiểm thử dùng kỹ thuật phân vùng tương đương

Câu hỏi 2.23: Lãi suất tiền gửi theo năm của khách hàng cá nhân tại ngân hàng X được cho ở bảng dưới. Lãi được tính trên số ngày thực tế.

Kỳ hạn	VND	EUR	USD
Tiết kiệm			
Không kỳ hạn	1.20 %	0.01 %	0.10 %
7 ngày	1.20 %		
14 ngày	1.20 %		

1 tháng	5.00 %	0.10 %	1.20 %
2 tháng	6.50 %	0.10 %	1.20 %
3 tháng	6.80 %	0.20 %	1.20 %
6 tháng	7.00 %	0.30 %	1.20 %
9 tháng	7.00 %	0.40 %	1.20 %
12 tháng	7.50 %	0.50 %	1.20 %
24 tháng	8.00 %	0.80 %	1.20 %

(a) Thiết kế các ca kiểm thử dùng kỹ thuật phân tích giá trị biên
(b) Thiết kế các ca kiểm thử dùng kỹ thuật phân vùng tương đương

Câu hỏi 2.24: Một chương trình phân loại tam giác đọc vào các giá trị số nguyên trong khoảng [0,100]. 3 giá trị A, B, và C được dùng để biểu diễn độ dài 3 cạnh tam giác. Chương trình in ra thông báo 3 giá trị này có phải là 3 cạnh tam giác không.
(a) Thiết kế các ca kiểm thử dùng kỹ thuật phân tích giá trị biên
(b) Thiết kế các ca kiểm thử dùng kỹ thuật phân vùng tương đương

Câu hỏi 2.25: Một chương trình phân loại tam giác đọc vào các giá trị số nguyên trong khoảng [0,100]. 3 giá trị A, B, và C được dùng để biểu diễn độ dài 3 cạnh tam giác. Chương trình in ra thông báo 3 giá trị này có phải là 3 cạnh tam giác cân không.
(a) Thiết kế các ca kiểm thử dùng kỹ thuật phân tích giá trị biên
(b) Thiết kế các ca kiểm thử dùng kỹ thuật phân vùng tương đương

Câu hỏi 2.26: Một chương trình phân loại tam giác đọc vào các giá trị số nguyên trong khoảng [0,100]. 3 giá trị A, B, và C được dùng để biểu diễn độ dài 3 cạnh tam giác. Chương trình in ra thông báo 3 giá trị này có phải là 3 cạnh tam giác đều không.
(a) Thiết kế các ca kiểm thử dùng kỹ thuật phân tích giá trị biên
(b) Thiết kế các ca kiểm thử dùng kỹ thuật phân vùng tương đương

Câu hỏi 2.27: Một chương trình phân loại tam giác đọc vào các giá trị số nguyên trong khoảng [0,100]. 3 giá trị A, B, và C được dùng để biểu diễn độ dài 3 cạnh tam giác. Chương trình in ra thông báo 3 giá trị này có phải là 3 cạnh tam giác vuông không.
(a) Thiết kế các ca kiểm thử dùng kỹ thuật phân tích giá trị biên
(b) Thiết kế các ca kiểm thử dùng kỹ thuật phân vùng tương đương

- **Câu hỏi loại 3 điểm**

Câu hỏi 3.1: Cho hàm tìm kiếm nhị phân viết bằng C. input array v đã được sắp xếp theo giá trị tăng dần, n là kích thước mảng, ta cần tìm chỉ số mảng của phần tử x. Nếu không tìm thấy x trong mảng, trả về giá trị -1.

```
int binSearch(int x, int v[], int n){
   int low, high, mid;
   low = 0;
```

```
    high = n - 1;
    while (low<=high){
        mid = (low + high)/2;
        if (x<v[mid])
            high = mid - 1;
        else if (x > v[mid])
            low = mid + 1;
        else
            return mid;
    }
    return -1;
}
```

a. Vẽ đồ thị luồng điều khiển
b. Từ đồ thị luồng điều khiển, xác định tập các đường từ đầu vào tới đầu ra để bao phủ được toàn bộ câu lệnh
c. Bổ xung thêm đường (nếu cần) để bao phủ hết các ngã rẽ (branch)
d. Với mỗi đường xác định ở trên, tìm biểu thức tiền tố tương ứng
e. Giải biểu thức tiền tố trên để sinh ra các đầu vào ca kiểm thử và sau đó ước lượng đầu ra tương ứng
f. Liệu tất cả các đường trên có khả thi hay không? Nếu không chỉ ra những đường không khả thi.

Câu hỏi 3.2: Giả mã bên dưới tính tổng các phần tử >0 của mảng a

```
sum_of_all_positive_numbers(a, num_of_entries, sum)
        sum =0;
        init = 1;
        while(init <= num_of_entries)
           if a[init] > 0
             sum = sum + a[init]
          endif
          init = init + 1
        endwhile
end sum_of_all_positive_numbers
```

a. Vẽ đồ thị luồng điều khiển
b. Từ đồ thị luồng điều khiển, xác định tập các đường từ đầu vào tới đầu ra để bao phủ được toàn bộ câu lệnh
c. Bổ xung thêm đường (nếu cần) để bao phủ hết các ngã rẽ (branch)
d. Với mỗi đường xác định ở trên, tìm biểu thức tiền tố tương ứng
e. Giải biểu thức tiền tố trên để sinh ra đầu vào các ca kiểm thử và sau đó ước lượng đầu ra tương ứng
f. Liệu tất cả các đường trên có khả thi hay không? Nếu không chỉ ra những đường không khả thi.

Câu hỏi 3.3: Hàm bên dưới trả về chỉ số phần tử cuối cùng trong x có giá trị bằng y. Nếu không tồn tại, trả về giá trị -1.

```
int findLast(int[] x, int y){
   for (int i = x.length -1; i > 0; i--){
      if (x[i] == y)
         return i;
   }
   return -1;
}
```

a. Vẽ đồ thị luồng điều khiển
b. Từ đồ thị luồng điều khiển, xác định tập các đường từ đầu vào tới đầu ra để bao phủ được toàn bộ câu lệnh
c. Bổ xung thêm đường (nếu cần) để bao phủ hết các ngã rẽ (branch)
d. Với mỗi đường xác định ở trên, tìm biểu thức tiền tố tương ứng
e. Giải biểu thức tiền tố trên để sinh ra đầu vào các ca kiểm thử và sau đó ước lượng đầu ra tương ứng
f. Liệu tất cả các đường trên có khả thi hay không? Nếu không chỉ ra những đường không khả thi.

Câu hỏi 3.4: Hàm bên dưới trả về chỉ số phần tử cuối cùng trong x có giá trị bằng 0. Nếu không tồn tại, trả về giá trị -1.

```
int lastZero(int[] x){
   for (int i = 0; i < x.length; i++){
      if (x[i] == 0)
         return i;
   }
   return -1;
}
```

a. Vẽ đồ thị luồng điều khiển
b. Từ đồ thị luồng điều khiển, xác định tập các đường từ đầu vào tới đầu ra để bao phủ được toàn bộ câu lệnh
c. Bổ xung thêm đường (nếu cần) để bao phủ hết các ngã rẽ (branch)
d. Với mỗi đường xác định ở trên, tìm biểu thức tiền tố tương ứng
e. Giải biểu thức tiền tố trên để sinh ra đầu vào các ca kiểm thử và sau đó ước lượng đầu ra tương ứng
f. Liệu tất cả các đường trên có khả thi hay không? Nếu không chỉ ra những đường không khả thi.

Câu hỏi 3.5: Hàm bên dưới trả về số phần tử là số >0.

```
int countPositive(int[] x){
   int count = 0;
   for (int i = 0 ; i < x.length; i++){
      if (x[i] >=0)
         count++;
   }
   return count;
}
```

a. Vẽ đồ thị luồng điều khiển
b. Từ đồ thị luồng điều khiển, xác định tập các đường từ đầu vào tới đầu ra để bao phủ được toàn bộ câu lệnh

c. Bổ xung thêm đường (nếu cần) để bao phủ hết các ngã rẽ (branch)
d. Với mỗi đường xác định ở trên, tìm biểu thức tiền tố tương ứng
e. Giải biểu thức tiền tố trên để sinh ra đầu vào các ca kiểm thử và sau đó ước lượng đầu ra tương ứng
f. Liệu tất cả các đường trên có khả thi hay không? Nếu không chỉ ra những đường không khả thi.

Câu hỏi 3.6: Cho đoạn code

```
public static void f1 (int x, int y) {
        if (x < y) { f2 (y); }
        else { f3 (y); };
}
public static void f2 (int a) {
        if (a % 2 == 0) {
                f3 (2*a);
        };
}
public static void f3 (int b) {
        if (b > 0) { f4(); }
        else { f5(); };
}
public static void f4()
        {... f6()....}
public static void f5()
        {... f6()....}
 public static void f6()
        {...}
```

a. Vẽ đồ thị luồng điều khiển
b. Từ đồ thị luồng điều khiển, xác định tập các đường từ đầu vào tới đầu ra để bao phủ được toàn bộ câu lệnh
c. Bổ xung thêm đường (nếu cần) để bao phủ hết các ngã rẽ (branch)
d. Với mỗi đường xác định ở trên, tìm biểu thức tiền tố tương ứng
e. Giải biểu thức tiền tố trên để sinh ra đầu vào các ca kiểm thử và sau đó ước lượng đầu ra tương ứng
f. Liệu tất cả các đường trên có khả thi hay không? Nếu không chỉ ra những đường không khả thi.

Câu hỏi 3.7: Cho hàm tìm kiếm nhị phân viết bằng C. Input array v đã được sắp xếp theo giá trị tăng dần, n là kích thước mảng, ta cần tìm chỉ số mảng của phần tử x. Nếu không tìm thấy x trong mảng, trả về giá trị -1.

```
int binSearch(int x, int v[], int n){
   int low, high, mid;
   low = 0;
   high = n - 1;
   while (low<=high){
      mid = (low + high)/2;
      if (x<v[mid])
         high = mid - 1;
```

```
        else if (x > v[mid])
            low = mid + 1;
        else
            return mid;
    }
    return -1;
}
```

Cho đầu vào các ca kiểm thử dưới đây:
t1= (x=1, v={1,2,5,7,9},n=5)
t2= (x=3, v={1,3,9},n=3)
t3= (x=9, v={1,2,5,7,9},n=5)
t4= (x=4, v={1,2,5,7,9},n=5)
t5= (x=10, v={1,2,5,7,9},n=5)
a. Vẽ đồ thị luồng điều khiển
b. Chỉ ra đường trên đồ thị luồng điều khiển tương ứng với mỗi đầu vào trên
c. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết câu lệnh
d. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết ngã rẽ
e. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết đường với n = 4
f. Liệu tất cả các đường tương ứng ở e có khả thi hay không? Nếu không chỉ ra những đường không khả thi.

Câu hỏi 3.8: Giả mã bên dưới tính tổng các số dương của mảng a

```
sum_of_all_positive_numbers(a, num_of_entries, sum)
        sum =0;
        init = 1;
        while(init < num_of_entries)
          if a[init] > 0
                sum = sum + a[init]
          endif
          init = init + 1
        endwhile
end sum_of_all_positive_numbers
```

Cho đầu vào ca kiểm thử dưới đây:
t1= (a={0,2,-5,7,-9}, num_of_entries=5)
t2= (a={1,-3,9}, num_of_entries=3)
t3= (a={1,-2,5,7,0}, num_of_entries=5)
t4= (a={-1,2,0,7,-9}, num_of_entries=5)
t5= (a={1,-2,5,7,9}, num_of_entries=5)
a. Vẽ đồ thị luồng điều khiển
b. Chỉ ra đường trên đồ thị luồng điều khiển tương ứng với mỗi đầu vào ca kiểm thử
c. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết câu lệnh
d. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết ngã rẽ
e. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết đường với num_of_entries = 4
f. Liệu tất cả các đường tương ứng ở e có khả thi hay không? Nếu không chỉ ra những đường không khả thi.

Câu hỏi 3.9: Hàm dưới đây tính số ngày giữa 2 ngày/tháng cho trước trong cùng 1 năm. Biết
1<=month1, month2<=12
1<=day1,day2<=31;
1<=year<=10000

```
public static int cal (int month1, int day1, int month2, int day2, int year) {
        int numDays;
        if (month2 == month1)
                numDays = day2 - day1;
        else {
                // bo qua thang thu  0.
                int daysIn[] = {0, 31, 0, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
                // kiem tra nam nhuan
                int m4 = year % 4;
                int m100 = year % 100;
                int m400 = year % 400;
                if ((m4 != 0) || ((m100 ==0) && (m400 != 0)))
                        daysIn[2] = 28;
                else
                        daysIn[2] = 29;
                // bắt đầu tính từ ngày
                numDays = day2 + (daysIn[month1] - day1);
                //cộng thêm số ngày ở khoảng giữa các tháng
                for (int i = month1 + 1; i <= month2-1; i++)
                        numDays = daysIn[i] + numDays;
                }
        return (numDays);
}
```

Cho đầu vào ca kiểm thử dưới đây:
t1= (12,20,12,30,2013)
t2= (2,10,11,1,2013)
t3= (1,1,12,1,2004)
t4= (2,20,3,1,2004)
t5= (2,20,3,1,1900}

a. Vẽ đồ thị luồng điều khiển
b. Chỉ ra đường trên đồ thị luồng điều khiển tương ứng với mỗi đầu vào ca kiểm thử
c. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết câu lệnh
d. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết ngã rẽ
e. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết đường với month1 = 2
f. Liệu tất cả các đường tương ứng ở e có khả thi hay không? Nếu không chỉ ra những đường không khả thi.

Câu hỏi 3.10: Hàm bên dưới trả về chỉ số phần tử cuối cùng trong x có giá trị bằng y. Nếu không tồn tại, trả về giá trị -1.

```
int findLast(int[] x, int y){
   for (int i = x.length -1; i > 0; i--){
      if (x[i] == y)
```

```
            return i;
    }
    return -1;
}
```

Cho đầu vào ca kiểm thử dưới đây:
t1= (x={5}, y=5)
t2= (x={1,-3,5}, y=5)
t3= (x={5,-2,5,7,0}, y=5)
t4= (x={-1,2,0,5,-9}, y=5)
t5= (x={1,-2,3,7,9}, y=5)
a. Vẽ đồ thị luồng điều khiển
b. Chỉ ra đường trên đồ thị luồng điều khiển tương ứng với mỗi đầu vào ca kiểm thử
c. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết câu lệnh
d. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết ngã rẽ
e. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết đường với số phần tử của mảng x = 4
f. Liệu tất cả các đường tương ứng ở e có khả thi hay không? Nếu không chỉ ra những đường không khả thi.

Câu hỏi 3.11: Hàm bên dưới trả về chỉ số phần tử cuối cùng trong x có giá trị bằng 0. Nếu không tồn tại, trả về giá trị -1.

```
int lastZero(int[] x){
    for (int i = 0; i < x.length; i++){
        if (x[i] == 0)
            return i;
    }
    return -1;
}
```

Cho đầu vào ca kiểm thử dưới đây:
t1= (x={5})
t2= (x={0})
t3= (x={5,-2,5,7,0})
t4= (x={-1,2,0,5,-9})
t5= (x={0,-2,3,7,9})
a. Vẽ đồ thị luồng điều khiển
b. Chỉ ra đường trên đồ thị luồng điều khiển tương ứng với mỗi đầu vào ca kiểm thử
c. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết câu lệnh
d. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết ngã rẽ
e. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết đường với số phần tử của mảng x = 4
f. Liệu tất cả các đường tương ứng ở e có khả thi hay không? Nếu không chỉ ra những đường không khả thi.

Câu hỏi 3.12: Hàm bên dưới trả về số phần tử là số >0.

```
int countPositive(int[] x){
    int count = 0;
```

```
   for (int i = 0 ; i < x.length; i++){
      if (x[i] >=0)
         count++;
   }
   return count;
}
```

Cho đầu vào ca kiểm thử dưới đây:

t1= (x={5})
t2= (x={0})
t3= (x={5,-2,5,7,0})
t4= (x={-1,2,0,5,-9})
t5= (x={0,-2,3,7,9})

a. Vẽ đồ thị luồng điều khiển
b. Chỉ ra đường trên đồ thị luồng điều khiển tương ứng với mỗi đầu vào ca kiểm thử
c. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết câu lệnh
d. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết ngã rẽ
e. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết đường với số phần tử của mảng x = 4
f. Liệu tất cả các đường tương ứng ở e có khả thi hay không? Nếu không chỉ ra những đường không khả thi.

Câu hỏi 3.13: Phương thức printPrimes bên dưới tìm và in ra n số nguyên tố.

```
private static void printPrimes (int n)
{
        int curPrime;
        int numPrimes;
        boolean isPrime;
        int [] primes = new int [MAXPRIMES];
        primes [0] = 2;
        numPrimes = 1;
        curPrime = 2;
        while (numPrimes < n)
        {
                curPrime++;
                isPrime = true;
                for (int i = 0; i <= numPrimes-1; i++)
                {
                        if (isDivisible (primes[i], curPrime))
                        {
                                isPrime = false;
                                 break;
                        }
                }
                if (isPrime)
                {
                        primes[numPrimes] = curPrime;
                        numPrimes++;
                }
```

```
        }

        for (int i = 0; i <= numPrimes-1; i++)
        {
        System.out.println ("Prime: " + primes[i]);
        }
        }
```

Cho đầu vào ca kiểm thử dưới đây:
t1= (n=0)
t2= (n=1)
t3= (n=2)
t4= (n=3)
a. Vẽ đồ thị luồng điều khiển
b. Chỉ ra đường trên đồ thị luồng điều khiển tương ứng với mỗi đầu vào ca kiểm thử
c. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết câu lệnh
d. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết ngã rẽ

Câu hỏi 3.14: Cho đoạn code

```
public static void f1 (int x, int y) {
        if (x < y) { f2 (y); }
        else { f3 (y); };
}
public static void f2 (int a) {
        if (a % 2 == 0) {
                f3 (2*a);
        };
}
public static void f3 (int b) {
        if (b > 0) { f4(); }
        else { f5(); };
}
public static void f4()
        {... f6()....}
public static void f5()
        {... f6()....}
 public static void f6()
        {...}
```

Sử dụng đầu vào ca kiểm thử dưới đây:
t1= f1(0, 0)
t2= f1(1, 1)
t3= f1(0, 1)
t4= f1(3, 2)
t5= f1(3, 4)
a. Vẽ đồ thị luồng điều khiển
b. Chỉ ra đường trên đồ thị luồng điều khiển tương ứng với mỗi đầu vào ca kiểm thử
c. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết nút
d. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết cạnh
e. Tìm tập đầu vào ca kiểm thử nhỏ nhất để bao phủ hết đường

Câu hỏi 3.15: public void foo2(int a, int b, int x) {

```
  if (a>1 && b==0) {
        x=x/a;
  }
  for ( int i = 1; i < 3; i++){
        if true x=x+1;
  }
}
```

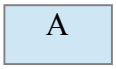
a. Vẽ đồ thị luồng điều khiển
b. Từ đồ thị luồng điều khiển, xác định tập các đường từ đầu vào tới đầu ra để bao phủ được toàn bộ ngã rẽ (branch)
c. Liệu tất cả các đường trên có khả thi hay không? Nếu không chỉ ra những đường không khả thi.
d. Xác định test case tương ứng với các đường khả thi

Câu hỏi 3.16:

```
sum(a, numEntry,sum){
                sum = 0;
                for (init =1; init <=numEntry;init++)
                        if (a[init]>0)
                                sum = sum + a[init];
                if (false)
                        sum = 0;
}
```

a. Vẽ đồ thị luồng điều khiển
b. Từ đồ thị luồng điều khiển, xác định tập các đường từ đầu vào tới đầu ra để bao phủ được toàn bộ ngã rẽ (branch)
c. Liệu tất cả các đường trên có khả thi hay không? Nếu không chỉ ra những đường không khả thi.
d. Xác định test case tương ứng với các đường khả thi

Câu hỏi 3.17: Cho sơ đồ gọi các module như sau



a. Xác định thứ tự tích hợp và các stub/driver (nếu cần) khi dùng kỹ thuật topdown
b. Xác định thứ tự tích hợp và các stub/driver (nếu cần) khi dùng kỹ thuật bottom up
c. Xác định thứ tự tích hợp và các stub/driver (nếu cần) khi dùng kỹ thuật sandwich

Câu hỏi 3.18: Cho code

```
int modifiedbinsearch(int X, int V[], int n){
  int low, high, mid;
  low = 0;
  high = n - 1;
  while (low <= high) {
   mid = (low + high)/2;
   if (X < V[mid]) {
        high = mid - 1;
        mid = mid - 1; }
   else if (X > V[mid])
         low = mid + 1;
          else
        return mid;
        }
   return -1; }
```

a. Vẽ đồ thị luồng dữ liệu.
b. Xác định điểm bất thường (anomaly) của code trên
c. Giả sử mảng V[] có ít nhất 1 phần tử, xác định đường không khả thi. Tìm test case tương ứng?

Câu hỏi 3.19: Chương trình SquaresLoopRange(*start-number, stop-number*) hiển thị bình phương của 1 dãy số từ *start-number* tới *stop-number*. Nếu *start-number* lớn hơn *stop-number* error message cần được hiển thị: Start-limit greater than stop-limit!Sq
a. Viết chương trình
b. Thiết kế test cases
c. Viết code JUnit tương ứng
a
a.
Câu hỏi 3.20: Viết chương trình MultiplesLoopRange(*start-number, stop-number, num*) hiển thị dãy số trong khoảng [*start-number, stop-number*] và dãy số phải là bội số của num. Nếu *start-number* lớn hơn *stop-number*, chương trình sẽ hiển thị dãy giảm dần.
a. Viết chương trình
b. Thiết kế test cases
c. Viết code JUnit tương ứng

Câu hỏi 3.21:
Hệ thống ghi lại nhật ký nhiệt độ theo thời gian. Nhưng output ở một format riêng, bao gồm một dãy các symbols, đầu tiên là 1 số biểu diễn nhiệt độ bắt đầu, ký hiệu tiếp biểu diễn sự thay đổi nhiệt độ so với trước đó. Các symbols được giải mã như sau:

- '.' không thay đổi
- '+' tăng 1 độ so với trước nó
- '-' giảm 1 độ so với trước nó

Các giá trị được biên dịch thành các số kiểu int.
Ta cần tính median của dữ liệu nhiệt độ. Đầu tiên, ta cần sắp xếp. Sau đó:

- Nếu mảng chứa số lẻ phần tử n, median là phần tử chính giữa:phần tử thứ $(n+1)/2$.

- Nếu mảng chứa số chẵn phần tử, median là giá trị trung bình của 2 phần tử thứ $n/2$ và $(n/2)+1$.

Lưu ý: nhiệt độ là integer, nhưng giá trị median là float.
a. Viết chương trình TempMedian
b. Thiết kế test cases
c. Viết code JUnit tương ứng

Câu hỏi 3.22:
DNA được tạo bởi 2 *DNA-Strands (chuỗi)*, chúng xoắn với nhau tạo thành 1 *double helix*.
Mỗi chuỗi DNA là chuỗi các bases. Gồm 4 bases:

- adenine (abbreviated A)
- cytosine (C)
- guanine (G)
- thymine (T)

Bases có cặp: A bắt cặp với T, C bắt cặp với G. Ví dụ, một sợi là A-C-G-G-T-C
Sợi còn lại sẽ là : T-G-C-C-A-G
Vì ta có các cặp A:T, C:G, G:C, G:C, T:A và C:G. Cách lưu trữ thông tin như vậy là dư thừa, nhưng ta có thể phân đôi 1 double helix và bỏ 1 phía đi, ta vẫn có thể tái tạo lại nó. Ngoài ra, nếu là muốn nhân đôi double helix, ta có thể chia thành 2 chuỗi, tại tạo lại mỗi phía và sẽ thu được 2 bản copy từ bản gốc.
Ta cần tạo 1 class để biểu diễn 1 chuỗi DNA. API cho class DNAStrand:
public DNAStrand(String dna): khởi tạo
public boolean isValidDNA(): Trả về true nếu DNA là valid, nghĩa là chỉ có các ký tự hoa A, T, C, G và chứa ít nhất 1 ký tự.
public String complementWC(): Trả về *Watson Crick complement*, là chuỗi DNA bù – sợi còn lại trong double helix. Thay T bằng A, A bằng T, C bằng G và G bằng C.
public String palindromeWC(): Trả về *Watson Crick Palindrome*, chuỗi đảo của chuỗi DNA bù.
public boolean containsSequence(String seq): Trả về true nếu DNA chứa chuỗi con seq.
public String toString(): Trả về string DNA.
a. Viết chương trình TempMedian
b. Thiết kế test cases
c. Viết code JUnit tương ứng

Ghi chú: *Ký hiệu (mã) câu hỏi được quy định X.Y*
Trong đó: + X tương đương với số điểm câu hỏi (X chạy từ 1 đến 5)
+ Y là câu hỏi thứ Y (Y chạy từ 1 trở đi)

2. Đề xuất các phương án tổ hợp câu hỏi thi thành các đề thi (Nếu thấy cần thiết):

3 câu 1 điểm
1 câu 2 điểm lý thuyết (chọn từ 2.1 – 2.10)
1 câu 2 điểm bài tập (chọn từ 2.11 – 2.19)
1 câu 3 điểm

3. Hướng dẫn cần thiết khác:

Ngân hàng câu hỏi thi này đã được thông qua bộ môn và nhóm cán bộ giảng dạy học phần.

Hà Nội, ngày tháng năm 2013

Trưởng khoa	Trưởng bộ môn	Giảng viên chủ trì biên soạn