


THÀNH PHỐ HỒ CHÍ MINH – NĂM 2022

MỤC LỤC

BÀI 1: KỸ THUẬT XỬ LÝ MẢNG MỘT CHIỀU, CON TRỎ VÀ XỬ LÝ NGOẠI LỆ	Error! Bookmark not defined.
BÀI 2: CÁC GIẢI THUẬT TÌM KIẾM VÀ SẮP XẾP	Error! Bookmark not defined.
BÀI 3. MẢNG STRUCT & FILE.....	Error! Bookmark not defined.
BÀI 4: KỸ THUẬT XỬ LÝ MẢNG HAI CHIỀU.....	Error! Bookmark not defined.
BÀI 5. ÔN TẬP - KIỂM TRA LẦN 1	Error! Bookmark not defined.
BÀI 6. XỬ LÝ CHUỖI.....	2
BÀI 7. KỸ THUẬT ĐỆ QUY	9
BÀI 8. KỸ THUẬT ĐỆ QUY (tt).....	Error! Bookmark not defined.
BÀI 9. BÀI TẬP TỔNG HỢP	Error! Bookmark not defined.
BÀI 10. ÔN TẬP - KIỂM TRA LẦN 2.....	Error! Bookmark not defined.
PHỤ LỤC BÀI TẬP THỰC HÀNH NÂNG CAO.....	Error! Bookmark not defined.

<p>Trường ĐH CNTP TP. HCM</p> <p>Khoa Công nghệ thông tin</p> <p>Bộ môn Công nghệ phần mềm</p> <p>THỰC HÀNH KỸ THUẬT LẬP TRÌNH</p>	<p>BÀI 6. XỬ LÝ CHUỖI</p>	
--	----------------------------------	---

A. MỤC TIÊU:

- Phân tích các yêu cầu của bài toán.
- Cài đặt được các hàm xử lý chuỗi ký tự cơ bản và sử dụng các hàm trong thư viện **string.h** vào các bài toán.
- Áp dụng các kỹ thuật cài đặt xử lý trên chuỗi vào những bài toán thực tế.

B. DỤNG CỤ - THIẾT BỊ THỰC HÀNH CHO MỘT SV:

STT	Chủng loại – Quy cách vật tư	Số lượng	Đơn vị	Ghi chú
1	Computer	1	1	

C. NỘI DUNG THỰC HÀNH

I. Tóm tắt lý thuyết

1. Khái niệm

Chuỗi ký tự là một dãy các phân tử, mỗi phân tử là một ký tự.

Lưu ý: Chuỗi ký tự được kết thúc bằng ký tự **'\0'**, gọi là ký tự kết thúc chuỗi. Do đó khi khai báo độ dài của chuỗi luôn luôn khai báo dư một phân tử để chứa ký tự **'\0'**.

Ví dụ 1.1. `char s[5] = "CNTT";` //khai báo chuỗi có 5 phân tử kiểu char và gán dãy ký tự CNTT và chuỗi được lưu trữ như sau:

'C'	'N'	'T'	'T'	'\0'
s[0]	s[1]	s[2]	s[3]	s[4]

Lưu ý: Chuỗi rỗng là chuỗi chưa có ký tự nào trong mảng và được ký hiệu **"**.

2. Khai báo chuỗi

Để khai báo một chuỗi, ta có 2 cách khai báo sau:

Cách 1: Con trỏ hằng

char <Tên_chuỗi>[<Số ký tự tối đa của chuỗi>];

Ví dụ 2.1. `char hoten[50];`

Cách 2: Con trỏ

char *<Tên_chuỗi>;

Ví dụ 2.2. `char *hoten;`

Ta có thể vừa khai báo vừa gán giá trị cho chuỗi như sau:

Ví dụ 2.3. `char monhoc[50] = "Kỹ thuật lập trình";`
`char s[10] = {'K', 'T', 'L', 'T', '\0'};`

Lưu ý: Việc gán nhiều giá trị hằng chuỗi như việc sử dụng dấu ngoặc kép (“”) chỉ hợp lệ khi khởi tạo mảng, tức là lúc khai báo mảng. Các biểu thức trong chương trình như:

```
s = "Hello";  
s[] = "Hello";
```

là không hợp lệ, cả câu lệnh dưới đây cũng vậy:

```
s = {'K', 'T', 'L', 'T', '\0'};
```

Vì về trái của một lệnh gán chỉ có thể là một phần tử của mảng chứ không thể là cả mảng, chúng ta có thể gán một chuỗi kí tự cho một mảng kiểu char sử dụng một phương pháp như sau:

```
s[0] = 'K';      s[1] = 'T';      s[2] = 'L';      s[3] = 'T';      s[4] = '\0';
```

Nhưng đây không phải là một phương pháp thực tế. Để gán giá trị cho một chuỗi kí tự, chúng ta có thể sử dụng các hàm kiểu **strcpy**, hàm này được định nghĩa trong thư viện **string.h** và được giải thích cụ thể trong phần 3.

3. Các thao tác xử lý chuỗi

a. Nhập dữ liệu cho chuỗi

Sử dụng hàm `gets()` (thuộc thư viện `<stdio.h>`) được coi là cách thức đơn giản và phổ biến nhất để ta tiến hành nhập liệu cho chuỗi. Cú pháp, sử dụng hàm này như sau: `gets(Tên_chuỗi);`

Ví dụ 3.1. Nhập dữ liệu từ bàn phím cho chuỗi `s` có kích thước tối đa 30 ký tự:

```
char s[30];  
printf("\n Moi ban nhap mot chuoi: ");  
gets(s);
```

Ngoài ra, ta cũng có thể sử dụng hàm `scanf()` với chuỗi định dạng “%s” để nhập dữ liệu cho chuỗi. Tuy nhiên, cách nhập này không cho phép nhập chuỗi có chứa khoảng trắng.

b. Xuất dữ liệu cho chuỗi

Có 2 cách xuất chuỗi ra màn hình như sau:

Cách 1: Sử dụng hàm `printf` với đặc tả “%s”

```
char monhoc[50] = "Tin hoc co so A";  
printf("%s", monhoc);      //Không xuống dòng
```

Cách 2: Sử dụng hàm `puts`

```
char monhoc[50] = "Tin hoc co so A";  
puts(monhoc);      //Tự động xuống dòng
```

c. Các hàm xử lý chuỗi trong thư viện <string.h>

STT	TÊN HÀM	CHỨC NĂNG	VÍ DỤ
1	int strlen (char s[]);	Trả về độ dài của chuỗi s.	char *s = "Borland International"; printf("Do dai s: %d", strlen(s)); Kết quả: Do dai s: 21
2	strcpy (char dest[], char src[]);	Sao chép nội dung chuỗi src vào chuỗi dest.	char dest[10]; char *src = "abcdefghi"; strcpy(dest, src); printf("%s\n", dest); Kết quả: abcdefghi
3	strncpy (char dest[], char src[], int n);	Chép n ký tự từ chuỗi src sang chuỗi dest. Nếu chiều dài src < n thì hàm sẽ điền khoảng trắng cho đủ n ký tự vào dest.	char dest[4]; char *src = "abcdefghi"; strncpy(dest, src, 3); printf("%s\n", dest); Kết quả: abc
4	strcat (char s1[], char s2[]);	Nối chuỗi s2 vào chuỗi s1.	char *s1 = "Khoa"; char *s2 = "CNTT"; strcat(s1, s2); printf("%s\n", s1); Kết quả: Khoa CNTT
5	strncat (char s1[], char s2[], int n)	Nối n ký tự đầu tiên của chuỗi s2 vào chuỗi s1.	char *s1 = "Khoa"; char *s2 = "CNTT"; strncat(s1, s2, 2); printf("%s\n", s1); Kết quả: Khoa CN
6	int strcmp (char s1[], char s2[])	So sánh 2 chuỗi s1 và s2 theo nguyên tắc thứ tự từ điển. Phân biệt chữ hoa và thường. Trả về: • 0: nếu s1 bằng s2. • > 0: nếu s1 lớn hơn s2. • < 0: nếu s1 nhỏ hơn s2.	char *s1 = "abcd"; char *s2 = "abCD"; if(strcmp(s1, s2)==0) printf("Giống nhau"); else printf("Khác nhau"); Kết quả: Khác nhau
7	int strncmp (char s1[], char s2[], int n)	Tương tự như strcmp(), nhưng chỉ so sánh n ký tự đầu tiên của hai chuỗi.	char *s1 = "abcd"; char *s2 = "abef"; if(strncmp(s1, s2, 2)==0) printf("Giống nhau"); else printf("Khác nhau"); Kết quả: Giống nhau

STT	TÊN HÀM	CHỨC NĂNG	VÍ DỤ
8	int stricmp (char s1[], char s2[])	Tương tự như strcmp(), nhưng không phân biệt hoa thường.	char *s1 = "abcd"; char *s2 = "abCD"; if(stricmp(s1, s2)==0) printf("Giống nhau"); else printf("Khác nhau"); Kết quả: Giống nhau
9	int strnicmp (char s1[], char s2[], int n);	Tương tự như strcmp(), nhưng chỉ so sánh n ký tự đầu tiên của hai chuỗi.	char *s1 = "aBcd"; char *s2 = "Abef"; if(strnicmp(s1, s2, 2)==0) printf("Giống nhau"); else printf("Khác nhau"); Kết quả: Giống nhau
10	char *strchr (char s[], char c);	Tìm lần xuất hiện đầu tiên của ký tự c trong chuỗi s. Trả về: <ul style="list-style-type: none"> • NULL: nếu không có. • Địa chỉ c: nếu tìm thấy. 	char s[15]; char *ptr, c = 'm'; strcpy(s, "Vi du tim ky tu"); ptr = strchr(s, c); if (ptr) printf("Ky tu %c tai: %d", c, ptr-s); else printf("Khong tim thay"); Kết quả: Ky tu m tai: 8

II. Bài tập có hướng dẫn

Bài 1. Nhập vào một chuỗi từ bàn phím. Tính và xuất độ dài của chuỗi (không dùng hàm trong thư viện string.h).

Hướng dẫn:

Phân tích:

- Input: chuỗi s.
- Output: n (là chiều dài chuỗi s).

Chương trình mẫu:

```
#include <stdio.h>
#include <conio.h>
int length(char s[])
{
    int i = 0;
    while(s[i] != '\0')
        i++;
}
```

```

        return i;
    }
    void main()
    {
        //sinh viên khai báo chuỗi, gọi và in giá trị cho hàm length
        getch();
    }

```

Bài 2. Nhập vào 1 chuỗi và 1 ký tự c, kiểm tra ký tự c có trong chuỗi hay không, nếu có đưa ra số lần xuất hiện của ký tự đó trong chuỗi.

Hướng dẫn:

Phân tích:

- Input: chuỗi s và ký tự c.
- Output: Trả lời ký tự c có trong chuỗi s hay không.

Chương trình mẫu:

```

#include <stdio.h>
#include <conio.h>
#include <string.h>
int kiemTra(char s[], char c)
{
    int dem = 0;
    //for(int i = 0; i < strlen(s); i++) {
    for(int i = 0; s[i] != '\0'; i++)
    {
        if(s[i] == c)
            dem++;
    }
    return dem;
}
void main()
{
    /*sinh viên khai báo chuỗi và 1 ký tự, gọi và in giá trị cho hàm kiemtra*/
    getch();
}

```

Bài 3. Nhập vào một chuỗi từ bàn phím. Đếm có bao nhiêu từ trong chuỗi vừa nhập (giả sử chuỗi nhập vào đúng chuẩn).

Hướng dẫn:

Phân tích:

- Input: chuỗi s.
- Output: n (là số từ của chuỗi s).

Chương trình mẫu:

```
#include <stdio.h>
#include <conio.h>
#include <string.h>

int demTu(char s[])
{
    int dem = 0;
    //for(int i = 0; i < strlen(s); i++) {
    for(int i = 0; s[i] != '\0'; i++)
    {
        if(s[i] == ' ')
            dem++;
    }
    return dem;
}

void main()
{
    //sinh viên tự viết code gọi hàm và xuất kết quả
    getch();
}
```

III. Bài thực hành trên lớp

Bài 1. Kiểm tra xem chuỗi s có chứa toàn ký số hay không?

Bài 2. Viết hàm đổi những ký tự đầu tiên của mỗi từ thành chữ in hoa và những từ không phải đầu câu sang chữ thường.

Bài 3. Viết hàm xóa những khoảng trắng thừa trong chuỗi (hay còn gọi là chuẩn hóa chuỗi).

Bài 4. Viết hàm tìm kiếm tên trong một chuỗi họ tên. Nếu có thì xuất ra là tên chuỗi họ tên, ngược lại thông báo tên không tồn tại.

Bài 5. Viết hàm cắt chuỗi họ tên thành 2 chuỗi con: họ lót và tên.

Ví dụ: chuỗi họ tên là: “Nguyễn Văn Anh” cắt ra 2 chuỗi là chuỗi họ lót: “Nguyễn Văn”, chuỗi tên là: “Anh”.

Bài 6. Nhập vào một danh sách sinh viên và hiển thị danh sách sinh viên ra màn hình.

Áp dụng giải thuật Brute Force, tìm và xuất vị trí các chuỗi P trong T,

a) Với T là chuỗi nhập vào từ bàn phím.

b) Với T là chuỗi văn bản đọc từ file text.

IV. Bài thực hành nâng cao

Bài 7. Kiểm tra xem chuỗi nhập vào có đối xứng hay không?

Bài 8. Nhập vào chuỗi s1 và s2, cho biết vị trí xuất hiện của chuỗi s2 trong s1 (nếu có). Nếu s2 không có trong s1, thực hiện nối s2 vào cuối s1.

Bài 9. Nhập vào chuỗi str, chuỗi cần chèn strInsert và vị trí cần chèn vt. Hãy chèn chuỗi strInser vào chuỗi str tại vị trí vt.

Bài 10. Nhập một chuỗi bất kì, yêu cầu nhập 1 kí tự muốn xóa. Thực hiện xóa tất cả những kí tự đó trong chuỗi.

Bài 11. Nhập vào 1 mảng các chuỗi. Tìm xuất ra màn hình những chuỗi chứa toàn ký tự số.

Bài 12. Cho 2 chuỗi s1 và s2 là 2 từ (chỉ chứa chữ cái hoặc ký số). Tính xem cần thực hiện bao nhiêu phép biến đổi để chuyển chuỗi s1 thành s2. Biết rằng có 3 phép biến đổi

- Thêm ký tự
- Xóa ký tự
- Đổi ký tự

Ví dụ: xét 2 từ s1= “Kitten” và s2=“Sitting”

Cần thực hiện 3 phép đổi để chuyển s1 thành s2. Đó là:

- Đổi ‘K’ thành ‘S’
- Đổi ‘e’ thành ‘i’
- Thêm ‘g’ vào cuối.

---HẾT---

<p>Trường ĐH CNTP TP. HCM</p> <p>Khoa Công nghệ thông tin</p> <p>Bộ môn Công nghệ phần mềm</p> <p>THỰC HÀNH KỸ THUẬT LẬP TRÌNH</p>	<p>BÀI 7.</p> <p>KỸ THUẬT ĐỆ QUY</p>	
--	--	--

A. MỤC TIÊU:

Thực hành kỹ thuật đệ quy dạng viết hàm cơ bản:

- Đệ quy tuyến tính
- Đệ quy nhị phân
- Đệ quy phi tuyến
- Đệ quy lồng
- Đệ quy tương hỗ

Thực hành kỹ thuật khử đệ quy.

B. DỤNG CỤ - THIẾT BỊ THỰC HÀNH CHO MỘT SV:

STT	Chủng loại – Quy cách vật tư	Số lượng	Đơn vị	Ghi chú
1	Computer	1	1	

C. NỘI DUNG THỰC HÀNH

I. Bài tập có hướng dẫn

1. **Đệ quy tuyến tính:** là hàm bên trong thân hàm có duy nhất một lời gọi đệ quy

Ví dụ 1: Viết hàm đệ quy tính X^n với X là số thực, n là số nguyên được xác định như sau:

$$X^n = \begin{cases} 1, & n = 0 \\ X * X^{n-1}, & n \neq 0 \end{cases}$$

```
float tinhLuyThua(float x, int n)
{
    float kq;
    if(n == 0)                //Điều kiện dừng
        kq = 1;
    else
        kq = x*tinhLuyThua(x, n-1);    //Gọi đệ quy
    return kq;
}
```

2. **Đệ quy nhị phân:** là hàm trong thân hàm có 2 lời gọi đệ quy

Ví dụ 2: Viết hàm đệ quy tính các tổ hợp chập k của n phần tử theo công thức sau:

$$C_n^k = \begin{cases} 1, & \text{nếu } k = 0 \text{ hoặc } k = n \\ C_{n-1}^{k-1} + C_{n-1}^k, & \text{nếu } 0 < k < n \end{cases}$$

```
int tinhToHop (int n, int k)
{
    if(k == 0 || k == n)
        return 1;    // Phần cơ sở
```

```
else
```

```
    return tinhToHop (n-1, k-1) + tinhToHop (n-1, k); //2 lần gọi hàm đệ qui
```

```
}
```

3. **Đệ qui phi tuyến:** là hàm có lời gọi hàm chính nó được đặt trong vòng lặp

Ví dụ 3: Viết hàm đệ qui tính $S(n)$ với n là số nguyên theo công thức sau:

$$S(n) = \begin{cases} 1, & \text{nếu } n = 1 \\ S(1) + S(2) + \dots + S(n-1), & \text{nếu } n > 1 \end{cases}$$

```
int Sn(int n)
```

```
{
```

```
    if(n == 1)
```

```
        return 1;
```

```
    int kq = 1;
```

```
    for (int i = 1; i < n; i++)
```

```
        kq += Sn(i); // hàm đệ qui được đặt trong vòng lặp
```

```
    return kq;
```

```
}
```

4. **Đệ qui lồng:** là hàm Tham số trong lời gọi đệ quy là một lời gọi đệ quy

Ví dụ 4: Viết hàm tính giá trị Ackermann's theo công thức sau:

$$Acker(m, n) = \begin{cases} n + 1, & \text{nếu } m = 0 \\ Acker(m - 1, 1), & \text{nếu } n = 0 \wedge m > 0 \\ Acker(m - 1, Acker(m, n - 1)), & \text{nếu } m > 0 \wedge n > 0 \end{cases}$$

Hàm số Ackermann là một hàm thực được mang tên nhà toán học người Đức Wilhelm Ackermann (1896–1962). Hàm Ackermann đôi khi còn được gọi là hàm Ackermann-Peter

```
int ackerman (int m, int n)
```

```
{
```

```
    if (m == 0)
```

```
        return (n+1);
```

```
    if (n == 0)
```

```
        return ackerman(m-1, 1);
```

```
    return ackerman(m-1, ackerman(m, n-1));
```

```
}
```

5. **Đệ qui tương hỗ:** Trong đệ quy tương hỗ có 2 hàm, và trong thân của hàm này có lời gọi của hàm kia, điều kiện dừng và giá trị trả về của cả hai hàm có thể giống nhau hoặc khác nhau.

Yêu cầu: Viết hàm đệ qui tính $X(n)$, $Y(n)$ với n là số nguyên theo công thức sau:

$$X(0) = Y(0) = 1$$

$$X(n) = X(n-1) + Y(n-1)$$

$$Y(n) = X(n-1) * Y(n-1)$$

```
int X(int n)
```

```
{
```

```
    if (n == 0)
```

```

        return 1;
    else
        return X(n-1) + Y(n-1);    //Hàm X(n) có lời gọi hàm đến Y(n)
}
int Y(int n)
{
    if(n == 0)
        return 1;
    else
        return X(n-1) * Y(n-1);    //Hàm Y(n) có lời gọi hàm đến X(n)
}

```

Khử đệ qui: Giải thuật giải bài toán bằng đệ quy thường gọn, dễ hiểu. Tuy nhiên, bài toán xử lý bằng giải thuật đệ quy thường tốn nhiều bộ nhớ và thời gian. Nên mọi giải thuật đệ qui đều thay thế bằng một giải thuật không đệ quy. Khử đệ qui để có chương trình không đệ quy. Có 2 cách khử đệ quy: khử đệ quy bằng vòng lặp, khử đệ quy bằng Stack

Yêu cầu: Viết hàm đệ qui tính X^n với X là số thực, n là số nguyên được xác định như sau:

$$X^n = \begin{cases} 1, & n = 0 \\ X * X^{n-1}, & n \neq 0 \end{cases}$$

```

float tinhLuyThua(float x, int n)
{
    float kq = 1;
    for(int i = 1; i <= n; i++)
    {
        kq *= x;
    }
    return kq;
}

```

II. Bài tập thực hành trên lớp

Bài 1. Viết hàm tính các biểu thức S(n) theo 2 cách đệ quy và khử đệ quy (nếu có thể), với n là số nguyên dương nhập từ bàn phím:

$$S(n) = 1 + 2 + 3 + \dots + n.$$

$$S(n) = \sqrt{5 + \sqrt{5 + \dots + \sqrt{5 + \sqrt{5}}}} \text{ có } n \text{ dấu căn.}$$

$$S(n) = \frac{1}{2} + \frac{2}{3} + \dots + \frac{n}{n+1}$$

$$S(n) = 1 + \frac{1}{3} + \frac{1}{5} + \dots + \frac{1}{2n+1}$$

$$S(n) = 1.2 + 2.3 + 3.4 + 4.5 + \dots + n.(n+1)$$

$$S(n) = \frac{1.2!}{2 + \sqrt{3}} + \frac{2.3!}{3 + \sqrt{4}} + \frac{3.4!}{4 + \sqrt{5}} + \dots + \frac{n.(n+1)!}{(n+1) + \sqrt{(n+2)}}$$

$$S(n) = \frac{1 + \sqrt{1+2}}{2 + \sqrt{3!}} + \frac{2 + \sqrt{2+3}}{3 + \sqrt{4!}} + \frac{3 + \sqrt{3+4}}{4 + \sqrt{5!}} + \dots + \frac{n + \sqrt{n+n+1}}{(n+1) + \sqrt{(n+2)!}}$$

Bài 2. Viết hàm tìm ước chung lớn nhất của 2 số nguyên dương a, b.

Gợi ý: Nếu $a > b$ thì $\text{UCLN}(a, b) = \text{UCLN}(b, a-b)$,

ngược lại $\text{UCLN}(a, b) = \text{UCLN}(a, b-a)$.

Bài 3. Viết hàm tìm giá trị phần tử thứ n của cấp số cộng có hạng đầu là a, công sai là r:

$$U_n = \begin{cases} a, & \text{nếu } n = 1 \\ U_{n-1} + r, & \text{nếu } n > 1 \end{cases}$$

Bài 4. Cho dãy số A_n theo các công thức quy nạp như sau, hãy viết chương trình tính số hạng thứ n, với n là số nguyên dương:

a. $A_0 = 1$; $A_1 = 0$; $A_2 = -1$; $A_n = 2A_{n-1} - 3A_{n-2} - A_{n-3}$.

b. $A_1 = 1$; $A_2 = 2$; $A_3 = 3$; $A_{n+3} = 2A_{n+2} + A_{n+1} - 3A_n$

Viết chương trình tính số hạng thứ n.

Bài 5. Cho dãy số x_n được định nghĩa như sau:

$$x_0 = 1 ; x_1 = 2 ; x_n = nx_0 + (n-1)x_1 + \dots + x_{n-1}$$

Viết hàm đệ quy tính x_n với $n \geq 0$.

Bài 6. Đếm số chữ số nguyên dương n.

Bài 7. Tìm số Fibonacci thứ n. Biết rằng:

$$\text{Fibonacci}(n) = \begin{cases} 1 & \text{với } n \leq 2 \\ \text{Fibonacci}(n-1) + \text{Fibonacci}(n-2) & \text{với } n > 2 \end{cases}$$

Bài 8. Xuất dãy số Fibonacci mà giá trị các số nhỏ hơn m.

III. Bài tập về nhà

Bài 9. Viết hàm tính các biểu thức $S(n)$ theo 2 cách đệ quy và khử đệ quy (nếu có thể), với n là số nguyên dương nhập từ bàn phím:

$$S(n) = \frac{1}{1.2.3} + \frac{1}{2.3.4} + \frac{1}{3.4.5} + \dots + \frac{1}{n.(n+1).(n+2)}$$

$$S(n) = 1^2 + 2^2 + \dots + n^2$$

$$S(n) = 1 + (1+2) + (1+2+3) + \dots + (1+2+3+\dots+n)$$

$$S(n) = -\frac{1+2}{2!} + \frac{3+4}{4!} - \frac{5+6}{6!} \dots + (-1)^n \frac{(2n-1) + (2n)}{(2n)!}$$

Bài 10. Viết hàm xuất dãy có số Fibonacci thuộc đoạn từ [m, n], biết rằng số Fibonacci là số có dạng:

$$\text{Fibonacci}(n) = \begin{cases} 1 & \text{với } n \leq 2 \\ \text{Fibonacci}(n-1) + \text{Fibonacci}(n-2) & \text{với } n > 2 \end{cases}$$

Ví dụ: nhập m = 5, n = 20 \rightarrow dãy số Fibonacci có 3 số thỏa là: 5 8 13

Bài 11. Viết hàm tìm số Fibonacci lớn nhất nhưng nhỏ hơn số nguyên dương n cho trước theo 2 cách đệ quy và khử đệ quy.

Ví dụ: nhập n = 15 \rightarrow số Fibonacci lớn nhất nhỏ hơn 15 là 13.

Bài 12. Viết hàm tính số hạng thứ n của 2 dãy sau:

$$x_0 = 1, y_0 = 0,$$

$$x_n = x_{n-1} + y_{n-1} \text{ với mọi } n > 0$$

$$y_n = 3x_{n-1} + 2y_{n-1} \text{ với mọi } n > 0$$

Bài 13. Viết hàm tìm giá trị phần tử thứ n của cấp số nhân có hạng đầu là a , công bội là q :

$$U_n = \begin{cases} a, & \text{nếu } n = 1 \\ qU_{n-1}, & \text{nếu } n > 1 \end{cases}$$

Bài 14. Viết hàm tính biểu thức $U(n)$ sau, với n là số nguyên dương nhập từ bàn phím:

$$U_n = \begin{cases} n, & \text{với } n < 6 \\ U_{n-5} + U_{n-4} + U_{n-3} + U_{n-2} + U_{n-1} & \text{với } n \geq 6 \end{cases}$$

Bài 15. Dãy An được cho như sau:

$$A_1 = 1 ;$$

$$A_n = n * (A_1 + A_2 + \dots + A_{n-1}).$$

Viết hàm tính A_n sử dụng kỹ thuật đệ quy.

Bài 16. Với mỗi $n \geq 1$, số Y_n được tính như sau:

$$Y_1 = 1 ; Y_2 = 2 ; Y_3 = 3 ; Y_n = Y_{n-1} + 2Y_{n-2} + 3Y_{n-3} \text{ nếu } n \geq 4$$

Viết hàm tính Y_n bằng 2 cách đệ quy.

Bài 17. Với mỗi $n \geq 1$, số X_n được tính như sau:

$$X_1 = 1 ; X_2 = 1 ; X_n = X_{n-1} + (n-1)X_{n-2} \text{ với } n \geq 3$$

Viết hàm tính X_n bằng cách đệ quy

Bài 18. Cho dãy số x_n được định nghĩa như sau:

$$x_0 = 1 ; x_1 = 2 ; x_n = nx_0 + (n-1)x_1 + \dots + x_{n-1}$$

Viết hàm đệ quy tính x_n với $n \geq 0$.

Bài 19. Dãy An được cho như sau:

$$A_1 = 1$$

$$A_{2n} = n + A_n + 2$$

$$A_{2n+1} = n^2 + A_n \cdot A_{n+1} + 1$$

Viết hàm đệ quy tính A_n

--HẾT--