

CHƯƠNG 4. PHƯƠNG THỨC (Method)

Cấu trúc chương trình

2



Ví dụ

3

Xét chương trình nhập vào số nguyên dương n , in ra màn hình các số nguyên tố nhỏ hơn n

Ví dụ:

Nhập $n = 10$

Kết quả in ra màn hình là: 2, 3, 5, 7

```
static void Main(string []args)
```

```
{ int n;
```

4

```
    Console.Write("Nhap so nguyen duong: ");
```

```
    n = int.Parse(Console.ReadLine());
```

```
    Console.WriteLine("Cac so nguyen to nho hon n la:");
```

```
    for(int so=2; so<n; so++)
```

```
    {
```

```
        int d=0;
```

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

```
        {
```

```
            if(so%i==0)
```

```
                d++;
```

```
        }
```

```
        if(d==2)
```

```
            Console.Write(so + " ");
```

```
    }
```

```
}
```

Kiểm tra xem **so** có phải
là số nguyên tố?

```
static void Main(string []args)
```

```
{    int n;
```

5

Nhập số nguyên dương n

```
Console.WriteLine("Cac so nguyen to nho hon n la:");
```

```
for(int so=2; so<n; so++)
```

```
{
```

Kiểm tra xem **so** có phải là số nguyên tố không? Nếu là số nguyên tố thì in **so** ra màn hình

```
}
```

```
}
```

```
static void NhapSoNguyen(out int n)
```

```
{
```

```
    Console.WriteLine("Nhap so nguyen duong: ");  
    n = int.Parse(Console.ReadLine());
```

```
}
```

```
static bool LaSNT(int k)
```

```
{
```

```
    int d=0;  
    for(int i=1; i<=k; i++)
```

```
{
```

```
        if(k%i==0)
```

```
            d++;
```

```
}
```

```
    return d==2;
```

```
}
```

Tham số

Tên hàm

```
static void Main(string []args)
```

```
7 {
```

```
    int n;
```

Gọi hàm

```
    NhapSoNguyen(out n);
```

Truyền đối số

```
    Console.WriteLine("Cac so nguyen to nho hon n la:");
```

```
    for(int so=2; so<n; so++)
```

```
    {
```

Gọi hàm

```
        if(LaSNT(so)==true)
```

Truyền đối số

```
        Console.Write(so + " ");
```

```
    }
```

```
}
```

Khái niệm

8

- Phương thức (hàm) là một đoạn chương trình độc lập **thực hiện trọn vẹn một công việc nhất định** và trả về kết quả cho phương thức gọi nó

Khi nào sử dụng phương thức?

- Khi có một công việc giống nhau cần thực hiện ở nhiều vị trí
- Khi cần chia nhỏ chương trình để dễ quản lý

Phương thức

Mẫu tổng quát của phương thức

<phạm vi> <KDL> TênPhươngThức([tham số]);

Phạm vi

- Xác định phạm vi hay cách phương thức được gọi (sử dụng)
- Các từ khoá phạm vi : private, public, static

Phương thức

KDL của phương thức (đầu ra), gồm 2 loại

- ▶ void: Không trả về giá trị
- ▶ float / int / long / string / kiểu cấu trúc / ... :
Trả về giá trị có KDL tương ứng với kết quả xử lý

Phương thức

- Tên phương thức : Đặt tên theo qui ước sao cho phản ánh đúng chức năng thực hiện của phương thức
- Danh sách các tham số (nếu có) : đầu vào của phương thức (trong một số trường hợp có thể là đầu vào và đầu ra của phương thức nếu kết quả đầu ra có nhiều giá trị - Tham số này gọi là tham chiếu)

Khi hàm xử lý biến toàn cục thì không cần tham số

static int a, b; _____

Nên hạn chế dùng

```
static void Nhap()
```

```
{
```

```
    Console.Write("Nhap a: ");
```

```
    a = int.Parse(Console.ReadLine());
```

```
    Console.Write("Nhap b: ");
```

```
    b = int.Parse(Console.ReadLine());
```

```
}
```

```
static void Xuat()
```

```
{
```

```
    Console.WriteLine("a = {0}; b = {1}", a, b);
```

```
}
```

```
static void Main(string[] args)
```

```
{
```

```
    Nhap();
```

```
    Xuat();
```

```
}
```

Phương thức không trả về giá trị

static void TênPhươngThức([danh sách các tham số])

{

Khai báo các biến cục bộ

Các câu lệnh hay lời gọi đến phương thức khác.

}

- Gọi: *TênPhươngThức*(danh sách tên các đối số);
- Những phương thức loại này thường rơi vào những nhóm chức năng: Nhập / xuất dữ liệu, thống kê, sắp xếp, liệt kê

Viết chương trình nhập số nguyên dương n và in ra màn hình các ước số của n

- Input: số nguyên dương (Xác định tham số)
 - Output: In ra các ước số của n (Xác định KDL trả về của phương thức)
 - Xuất → Không cần trả về giá trị → KDL là *void*.
 - Xác định tên phương thức: Phương thức này dùng in ra các US của n nên có thể đặt là LietKeUocSo
- static void LietKeUocSo(uint n)*

```
static void LietKeUocSo(uint n)
{
    for (int i = 1; i <= n; i++)
        if (n % i == 0)
            Console.Write("{0}\t", i);
}

static void Main(string[] args)
{
    uint n;
    Console.Write("Nhap so nguyen duong n: ");
    n=uint.Parse(Console.ReadLine());
    Console.Write("Cac uoc so cua {0}: ", n);
    LietKeUocSo(n);
    Console.ReadLine();
}
```


Phương thức có trả về kết quả

static <KDL> TênPhươngThức([tham số])

{

<KDL> kq;

Khai báo các biến cục bộ

Các câu lệnh hay lời gọi đến phương thức khác.

return kq;

}

Gọi:

<KDL> Tên biến = TênPhươngThức(tên các đối số);



Những phương thức này thường rơi vào các nhóm: *Tính tổng, tích, trung bình, đếm, kiểm tra, tìm kiếm*

Viết chương trình nhập số nguyên dương n và tính

$$S_n = 1 + 2 + 3 + \dots + n \quad ; n > 0$$

- Input: số nguyên dương n (Xác định tham số)
- Output: Tổng S (Xác định KDL trả về của phương thức)
 - Trả về giá trị tổng (S).
 - S là tổng các số nguyên dương nên S cũng là số nguyên dương \rightarrow Kiểu trả về của hàm là ulong.
- Xác định TênPhươngThức: Dùng tính tổng S nên có thể đặt là TongS

static ulong TongS(uint n)



```
static ulong TongS(uint n)
{
    ulong kq = 0;
    for (uint i = 1; i <= n; i++)
        kq += i;
    return kq;
}
static void Main(string[] args)
{
    ulong S;
    uint n;
    Console.Write("Nhap vao so nguyen n: ");
    n = uint.Parse(Console.ReadLine());
    S = TongS(n);
    Console.Write("Tong tu 1 den n: " + S);
    Console.ReadLine();
}
```

Bài tập – Nhập giá trị các biến trong Main()

- Viết chương trình tính diện tích và chu vi hình tròn.
- Nhập vào 3 số thực a, b, c và kiểm tra xem chúng có lập thành 3 cạnh của một tam giác hay không? Nếu có hãy tính diện tích, chiều dài mỗi đường cao của tam giác và in kết quả ra màn hình.

Diện tích tam giác: $s = \sqrt{p \cdot (p-a) \cdot (p-b) \cdot (p-c)}$

với p là nửa chu vi của tam giác

Tính các đường cao: $h_a = 2s/a$, $h_b = 2s/b$, $h_c = 2s/c$.

Bài tập

- Viết chương trình nhập 2 số nguyên dương a, b . Tìm USCLN & BSCNN.
- Viết chương trình nhập số nguyên dương n , tính tổng các ước số của n .

Ví dụ: Nhập $n=6$

Tổng các ước số từ 1 đến n : $1+2+3+6=12$.

- Nhập vào giờ, phút, giây. Kiểm tra xem giờ, phút, giây đó có hợp lệ hay không?

Tầm vực của biến

- Phạm vi khối
- Phạm vi hàm
- Phạm vi tập tin
- Phạm vi chương trình

- Một khối được giới hạn bởi ngoặc {}.
- Biến khai báo trong khối đó có phạm vi khối, nghĩa là nó chỉ hoạt động trong khối đó mà thôi. Phạm vi này còn gọi là **cục bộ**, và biến được gọi là **biến cục bộ**.

Phạm vi khối (tt)

```
static void Main(string args[])
{
    int i=20;
    {
        int i=10;
        Console.WriteLine("Gia tri i ben trong khoi: " + i);
    }
    Console.WriteLine("Gia tri i ben ngoai khoi: « + i);
}
```

Kết quả

Gia tri i ben trong khoi: 10

Gia tri i ben ngoai khoi: 20

Phạm vi hàm

Hoạt động từ đầu đến cuối một hàm, chỉ có tác dụng trong hàm

```
static void Main(string []args)
{
    int k;
    float m;
    double x;
    //Các lệnh khác
    //...
}
```


Biến được khai báo toàn cục và có kèm từ khóa **static**

```
static int x = 0;  
static int y = 0;  
static float z = 0.0;  
static void Main()  
{  
    int i;  
    //Các lệnh  
    .  
    .  
}
```

Phạm vi chương trình

- Được khai báo bên ngoài hàm – còn được gọi là biến toàn cục
- Không nên khai báo sử dụng nhiều nếu không thực sự cần thiết, vì nó sẽ gây trở ngại cho quá trình dò tìm lỗi khi debug chương trình

```
int a, b;  
static void Nhap()  
{  
    Console.Write("Nhap a: ");  
    a = int.Parse(Console.ReadLine());  
    Console.Write("Nhap b: ");  
    b = int.Parse(Console.ReadLine());  
}  
static void Main(string []args)  
{  
    int c;  
    Nhap();  
    c = a + b;  
    Console.WriteLine("Tong = "+c);  
}
```

Tham số là tham chiếu

- Tham số lưu kết quả xử lý của hàm: **out**
(thường dùng cho trường hợp nhập dữ liệu, kết quả hàm có nhiều giá trị)
- Tham số vừa làm đầu vào và đầu ra: **ref**
- Dùng từ khóa **ref** hoặc **out** trước KDL của khai báo tham số và trước tên đối số khi gọi phương thức.

Tham số là tham chiếu

Dùng từ khóa **ref** bắt buộc phải khởi gán giá trị ban đầu cho đối số trước khi truyền vào khi gọi phương thức (Nếu dùng **out** thì không cần thiết)

Hoán vị 2 số nguyên a, b cho trước

Đánh giá kết quả khi viết chương trình với hai trường hợp sau

1. Trường hợp không dùng tham chiếu
2. Trường hợp dùng tham chiếu: **ref**

Không dùng tham chiếu

```
static void HoanVi(int a, int b)
```

```
{
```

```
    int tam = a;
```

```
    a = b;
```

```
    b = tam;
```

```
    Console.WriteLine("Trong HoanVi: a = " + a + ";b = " + b);
```

```
}
```

```
static void Main(string[] args)
```

```
{
```

```
    int a = 5, b = 21;
```

```
    Console.WriteLine("Truoc HoanVi: a = {0}; b = {1}", a, b);
```

```
    HoanVi(a, b);
```

```
    Console.WriteLine("Sau HoanVi: a = " + a + ";b = " + b);
```

```
}
```

Dùng tham chiếu

```
static void HoanVi(ref int a, ref int b)
{
    int tam = a;
    a = b;
    b = tam;
    Console.WriteLine("Trong HoanVi: a = " + a + "; b = " + b);
}
static void Main(string[] args)
{
    int a = 5, b = 21;
    Console.WriteLine("Truoc HoanVi: a = {0}; b = {1}", a, b);
    HoanVi(ref a, ref b);
    Console.WriteLine("Sau HoanVi: a = " + a + "; b = " + b);
}
```

Ví dụ - sử dụng tham chiếu out

```
static void Nhap(out int a, out int b)
```

```
{  
    Console.Write("Nhap a: ");  
    a = int.Parse(Console.ReadLine());  
    Console.Write("Nhap b: ");  
    b = int.Parse(Console.ReadLine());  
}
```

```
static int Tong(int a, int b)
```

```
{  
    return a + b;  
}
```

```
static void Main(string[] args)
```

```
{  
    int a, b;  
    Nhap(out a, out b);  
    s=Tinh(a, b);  
    Console.WriteLine("{0}+{1}={2}", a, b, s);  
}
```


Nguyên tắc xây dựng hàm

33

➤ Kết quả của hàm? → KDL trả về của hàm

➤ Hàm làm gì? → Xác định tên hàm

➤ Cần những thông tin gì? → Tham số

Ứng với mỗi thông tin đã xác định, xác định xem đã có giá trị trước khi vào hàm chưa,

- Nếu chưa có → Tham chiếu

- Nếu có mà sau khi thực hiện xong hàm vẫn không thay đổi → Tham trị

- Nếu có mà sau khi thực hiện xong hàm thì giá trị cũng bị thay đổi theo → Tham chiếu

Bài tập

34

1. Viết chương trình tính diện tích và chu vi của hình chữ nhật với chiều dài và chiều rộng được nhập từ bàn phím.
2. Viết chương trình tính diện tích và chu vi hình tròn với bán kính được nhập từ bàn phím.
3. Nhập vào 3 số thực a , b , c và kiểm tra xem chúng có thành lập thành 3 cạnh của một tam giác hay không? Nếu có hãy tính diện tích, chiều dài mỗi đường cao của tam giác và in kết quả ra màn hình.
4. Viết chương trình nhập 2 số nguyên dương a , b . Tìm USCLN và BSCNN của hai số nguyên đó

Bài tập

35

5. Viết chương trình nhập số nguyên dương n , tính tổng các ước số dương của n .

Ví dụ: Nhập $n=6$

Tổng các ước số từ 1 đến n : $1+2+3+6=12$.

6. Nhập vào giờ, phút, giây. Kiểm tra xem giờ, phút, giây đó có hợp lệ hay không? In kết quả ra màn hình.

7. Viết chương trình nhập số nguyên dương n gồm k chữ số, đếm xem n có bao nhiêu chữ số là số nguyên tố.

8. Viết chương trình tính tiền thuê máy dịch vụ Internet. Với dữ liệu nhập vào là giờ bắt đầu thuê (GBD), giờ kết thúc thuê (GKT), số máy thuê (SoMay).

- Điều kiện cho dữ liệu nhập: $6 \leq \text{GBD} < \text{GKT} \leq 21$ (*Giờ là số nguyên*).
- Đơn giá: 2500đ cho mỗi giờ máy trước 17:30 và 3000đ cho mỗi giờ máy sau 17:30.

9. Viết chương trình tính tiền lương ngày cho công nhân, cho biết trước giờ vào ca, giờ ra ca của mỗi người.

Giả sử rằng:

- Tiền trả cho mỗi giờ trước 12 giờ: 6000đ và sau 12 giờ: 7500đ.
- Giờ vào ca sớm nhất là 6 giờ sáng và giờ ra ca trễ nhất là 18 giờ (*Giờ là số nguyên*).

Giới thiệu hàm đệ qui

- Một hàm được gọi có tính đệ qui nếu trong thân của hàm đó có lệnh gọi lại chính nó một cách tường minh hay tiềm ẩn.
- **Phân loại đệ qui**
 - Đệ qui tuyến tính.
 - Đệ qui nhị phân.
 - Đệ qui phi tuyến.
 - Đệ qui hỗ tương.

Đệ qui tuyến tính

Trong thân hàm có duy nhất một lời gọi hàm gọi lại chính nó một cách tường minh.

<Kiểu dữ liệu hàm> **TenHam** (<danh sách tham số>)

{

if (điều kiện dừng)

{

...

//Trả về giá trị hay kết thúc công việc

}

//Thực hiện một số công việc (nếu có)

... **TenHam** (<danh sách đối số>);

//Thực hiện một số công việc (nếu có)

}

Ví dụ: Tính $S(n) = 1 + 2 + 3 + \dots + n$

- Điều kiện dừng: $S(0) = 0$.

- Quy tắc (công thức) tính: $S(n) = S(n-1) + n$.

long TongS (int n)

{

if(n==0)

return 0;

return (TongS(n-1) + n);

}

Đề .qui nhị phân

Trong thân của hàm có hai lời gọi hàm gọi lại chính nó một cách tường minh.

<Kiểu dữ liệu hàm> **TenHam** (<danh sách tham số>)

```
{  
    if (điều kiện dừng)  
    {  
        ...  
        //Trả về giá trị hay kết thúc công việc  
    }  
    //Thực hiện một số công việc (nếu có)  
    ...TenHam (<danh sách đôi số>); //Giải quyết vấn đề nhỏ hơn  
    //Thực hiện một số công việc (nếu có)  
    ... TenHam (<danh sách đôi số>); //Giải quyết vấn đề còn lại  
    //Thực hiện một số công việc (nếu có)  
}
```


41

Ví dụ: Tính số hạng thứ n của dãy Fibonacci được định nghĩa như sau:

$$f_1 = f_0 = 1 ;$$

$$f_n = f_{n-1} + f_{n-2} ; \quad (n > 1)$$

Điều kiện dừng: $f(0) = f(1) = 1$.

long Fibonacci (int n)

{

if($n == 0$ // $n == 1$)

return 1;

return Fibonacci($n-1$) + Fibonacci($n-2$);

}

Đề .qui phi tuyển

42

Trong thân của hàm có lời gọi hàm gọi lại chính nó được đặt bên trong vòng lặp.

<Kiểu dữ liệu hàm> **TenHam** (<danh sách tham số>)

```
{  
    for (int i = 1; i<=n; i++)  
    {    //Thực hiện một số công việc (nếu có)  
        if (điều kiện dừng)  
        {  
            ...  
            //Trả về giá trị hay kết thúc công việc  
        }  
        else  
        {    //Thực hiện một số công việc (nếu có)  
            TenHam (<danh sách đối số>);  
        }  
    }  
}
```

Ví dụ: Tính số hạng thứ n của dãy $\{X_n\}$ được định nghĩa như sau:

$X_0 = 1$;

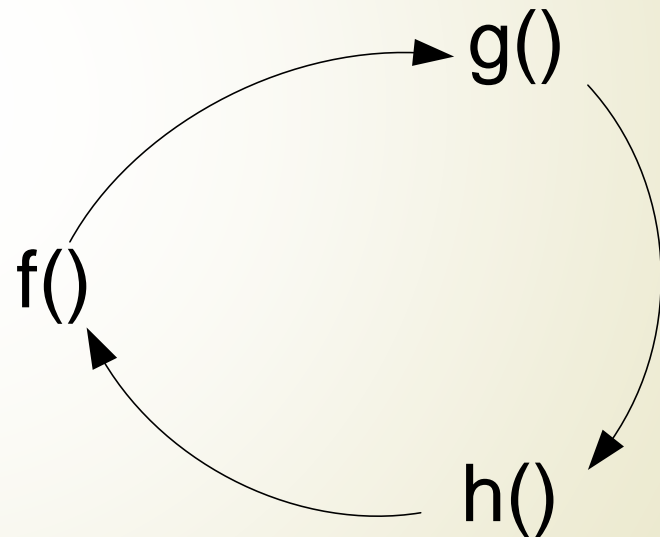
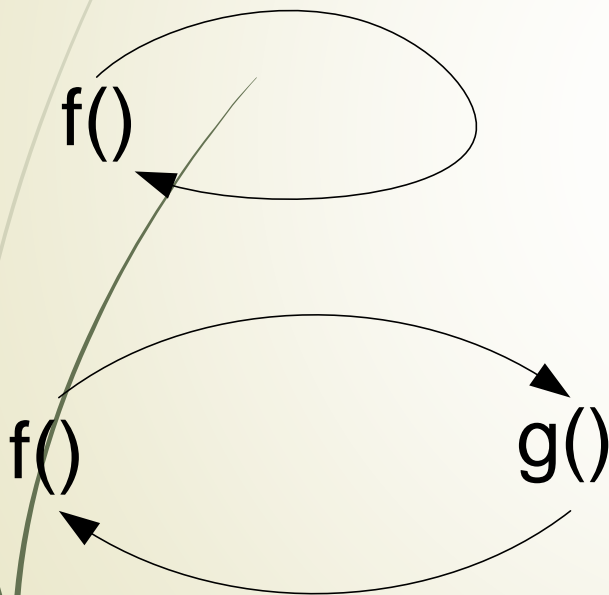
$$X_n = n^2 X_0 + (n-1)^2 X_1 + \dots + 1^2 X_{n-1} ; \quad (n \geq 1)$$

Điều kiện dừng: $X(0) = 1$.

```
long TinhXn (int n)
{
    if(n==0)
        return 1;
    long s = 0;
    for (int i=1; i<=n; i++)
        s = s + i * i * TinhXn(n-i);
    return s;
}
```

Đề .qui hũ tương

Trong thân của hàm này có lời gọi hàm đến hàm kia và trong thân của hàm kia có lời gọi hàm tới hàm này.



<Kiểu dữ liệu hàm> **TenHam2** (<danh sách tham số>);

45 <Kiểu dữ liệu hàm> **TenHam1** (<danh sách tham số>)

{

//Thực hiện một số công việc (nếu có)

...**TenHam2** (<danh sách tham số>);

//Thực hiện một số công việc (nếu có)

}

<Kiểu dữ liệu hàm> **TenHam2** (<danh sách tham số>)

{

//Thực hiện một số công việc (nếu có)

...**TenHam1** (<danh sách tham số>);

//Thực hiện một số công việc (nếu có)

}

Ví dụ: Tính số hạng thứ n của hai dãy $\{X_n\}$, $\{Y_n\}$ được định nghĩa như sau:

$$X_0 = Y_0 = 1;$$

$$X_n = X_{n-1} + Y_{n-1}; \quad (n > 0)$$

$$Y_n = n^2 X_{n-1} + Y_{n-1}; \quad (n > 0)$$

- Điều kiện dừng: $X(0) = Y(0) = 1$.

long TinhYn(int n);

long TinhXn (int n)

{

if(n==0)

return 1;

return TinhXn(n-1) + TinhYn(n-1);

}

long TinhYn (int n)

{

if(n==0)

return 1;

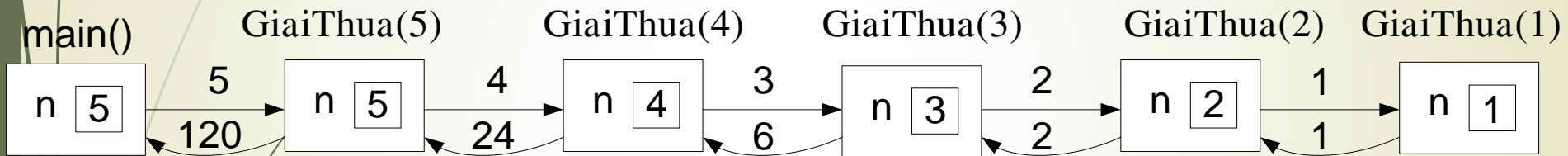
return n*n*TinhXn(n-1) + TinhYn(n-1);

}

Cách hoạt động hàm đệ quy

47

➡ Ví dụ tính $n!$ với $n=5$



Q&A

