TRƯỜNG ĐẠI HỌC THỦY LỢI
Khoa Công nghệ thông tin
Bộ môn Tin học và KTTT

NHẬP MÔN LẬP TRÌNH

Giảng viên: TS.GVC Bùi Thị Thanh Xuân

Email: xuanbtt@tlu.edu.vn

Điện thoại: 0902001581

Chương 5: Hàm

5.1. Khái niệm hàm

- Khái niệm chương trình con
- Phân loại: hàm và thủ tục

5.2. Khai báo và sử dụng hàm

- Khai báo và sử dụng

5.3. Phạm vi của biến

- Toàn cục và địa phương
- Biến static, biến register

5.4. Truyền tham số

- Truyền theo giá trị, truyền theo địa chỉ

Khái niệm chương trình con

- Chương trình con
  - Là đoạn chương trình thực hiện một nhiệm vụ cụ thể và được đóng gói theo cấu trúc xác định
  - Được định nghĩa một lần và sử dụng lại nhiều lần.
- Vai trò
  - Tổ chức một chương trình thành nhiều phần nhỏ để tăng tính cấu trúc.
  - Tiết kiệm thời gian bằng cách sử dụng chương trình con đã có thay vì viết lại.
  - Mở rộng tính năng cho chương trình bằng cách sử dụng các thư viện, ví dụ printf(), scanf()…
  - Gỡ lỗi, bảo trì chương trình dễ dàng hơn

Hàm trong ngôn ngữ C

- Các ngôn ngữ lập trình nói chung: 2 loại chương trình con
  - Hàm: thực hiện công việc và trả lại kết quả.
  - Thủ tục: thực hiện công việc, không trả lại kết quả.
- Ngôn ngữ C:
  - Chỉ có 1 loại chương trình con là hàm (function).
  - Hàm trong C tương đương cả hàm và thủ tục trong các ngôn ngữ khác.
  - Sử dụng kiểu void (kiểu dữ liệu không định kiểu) khi hàm không trả về dữ liệu.

Chương 5: Hàm

5.1. Khái niệm hàm
- Khái niệm chương trình con
- Phân loại: hàm và thủ tục

5.2. Khai báo và sử dụng hàm
- Khai báo và sử dụng

5.3. Phạm vi của biến
- Toàn cục và địa phương
- Biến static, biến register

5.4. Truyền tham số
- Truyền theo giá trị, truyền theo địa chỉ

Ví dụ

Khai báo chương trình con

Gọi chương trình con ra thực hiên

```
#include<stdio.h>
int bp(int x){
    int y;
    y =  x * x;
    return y;
}
int main(){
   int i;
   for (i=1; i< 20; i+=2)
      printf("%4d\n", bp(i));
   printf("\n");
   return 0;
}
```

```
   1
   9
  25
  49
  81
 121
 169
 225
 289
 361
```

Định nghĩa hàm

Cú pháp

Dòng đầu hàm

```
Kiểu_hàm Tên_hàm(DS khai báo tham số)
{
  [<Các khai báo cục bộ>]
  [<Các câu lệnh>]
}
```

Thân hàm

Dòng đầu hàm

Kiểu_hàm Tên_hàm(DS khai báo tham số)

- Mô tả các thông tin được trao đổi giữa bên trong và bên ngoài hàm.
  - Tên của hàm,
  - Các tham số đầu vào
    - Hàm cần những thông tin gì để hoạt động
  - Tham số đầu ra và giá trị trả về
    - Hàm cung cấp những thông tin gì cho môi trường
- Dùng phân biệt các hàm với nhau,
  - không tồn tại 2 hàm có dòng đầu hàm giống nhau.

Dòng đầu hàm→Tên hàm

Là tên do người sử dụng tự định nghĩa

- Tuân theo quy tắc đặt tên đối tượng
- Nên mang ý nghĩa gợi ý chức năng của hàm

Dòng đầu hàm→Khai báo các tham số hình thức

- Khai báo các thông tin cần cho hoạt động của hàm và các thông tin, kết quả tính toán được hàm trả lại.
  - Tham số chứa dữ liệu vào cung cấp cho hàm
  - Tham số chứa dữ liệu ra mà hàm tính toán được.
- Các tham số sử dụng trong khai báo hàm là tham số hình thức.
  - Nguyên tắc khai báo tham số hình thức như giống như khai báo một biến

    kiểu_dữ_liệu_của_tham_số tên_của_tham_số

Dòng đầu hàm→Khai báo các tham số hình thức

- Các tham số cung cấp cho hàm trong quá trình thực hiện hàm là tham số thực sự
  - Kiểu dữ liệu của tham số thực phải giống kiểu dữ liệu của tham số hình thức tương ứng với tham số thực sự đó,.
- Một hàm có thể có một, nhiều hoặc không có tham số nào cả
  - Nếu có nhiều tham số, phải được phân cách với nhau bằng dấu phẩy.
  - không có tham số vẫn phải có cặp dấu ngoặc đơn sau tên hàm

Dòng đầu hàm→Kiểu dữ liệu trả về

- Thông thường hàm sau khi được thực hiện sẽ trả về một giá trị kết quả tính toán nào đó.
- Để sử dụng được giá trị đó cần phải biết nó thuộc kiểu dữ liệu gì.
  - Kiểu dữ liệu của đối tượng tính toán được hàm trả về được gọi là kiểu dữ liệu trả về của hàm.

Dòng đầu hàm→Kiểu dữ liệu trả về

- Trong C, kiểu dữ liệu trả về của hàm có thể là kiểu dữ liệu bất kì (kiểu dữ liệu có sẵn hoặc kiểu dữ liệu do người dùng tự định nghĩa) nhưng không được là kiểu dữ liệu mảng.
- Nếu kiểu dữ liệu trả về là kiểu void thì hàm không trả về giá trị nào cả.
- Nếu không khai báo kiểu dữ liệu trả về thì chương trình dịch của C sẽ ngầm hiểu rằng kiểu dữ liệu trả về của hàm là kiểu int.

Thân hàm

- Danh sách các câu lệnh
- Thường có ít nhất một lệnh return

Hoạt động của hàm

- Thực hiện lần lượt các lệnh cho đến khi
  - Thực hiện xong tất cả các câu lệnh có trong thân hàm
  - Gặp lệnh return
    - Cú pháp chung

return [biểu_thức];

Thân hàm (tiếp)

Khi gặp lệnh *return biểu_thức*

- Tính toán giá trị của *biểu_thức*,
- Lấy kết quả tính toán được làm giá trị trả về cho lời gọi hàm
- Kết thúc việc thực hiện hàm, trở về chương trình đã gọi nó.

Nếu return không có phần *biểu_thức*,

- Kết thúc thực hiện hàm mà không trả về giá trị nào cả.
  - Dùng khi hàm được khai báo có kiểu trả về là void

Sử dụng hàm

Tên_hàm (DS_tham_số_thực _sự);

Ví dụ:

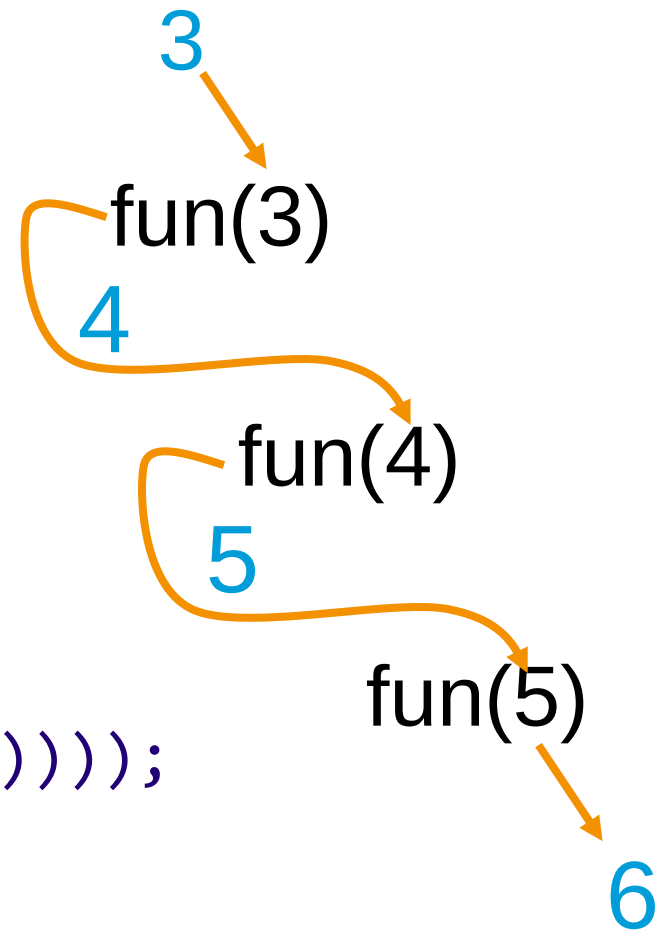
N = bp(1);N = bp(3);,…

Lưu ý:

- Gọi hàm thông qua tên hàm và các tham số được cung cấp thực sự cho hàm (*tham số thực sự*).
- Nếu hàm nhận nhiều tham số thì các tham số ngăn cách nhau bởi dấu phẩy
- Các tham số hình thức của hàm sẽ nhận các giá trị từ tham số truyền vào
- Sau khi thực hiện xong, trở về điểm mà hàm được gọi

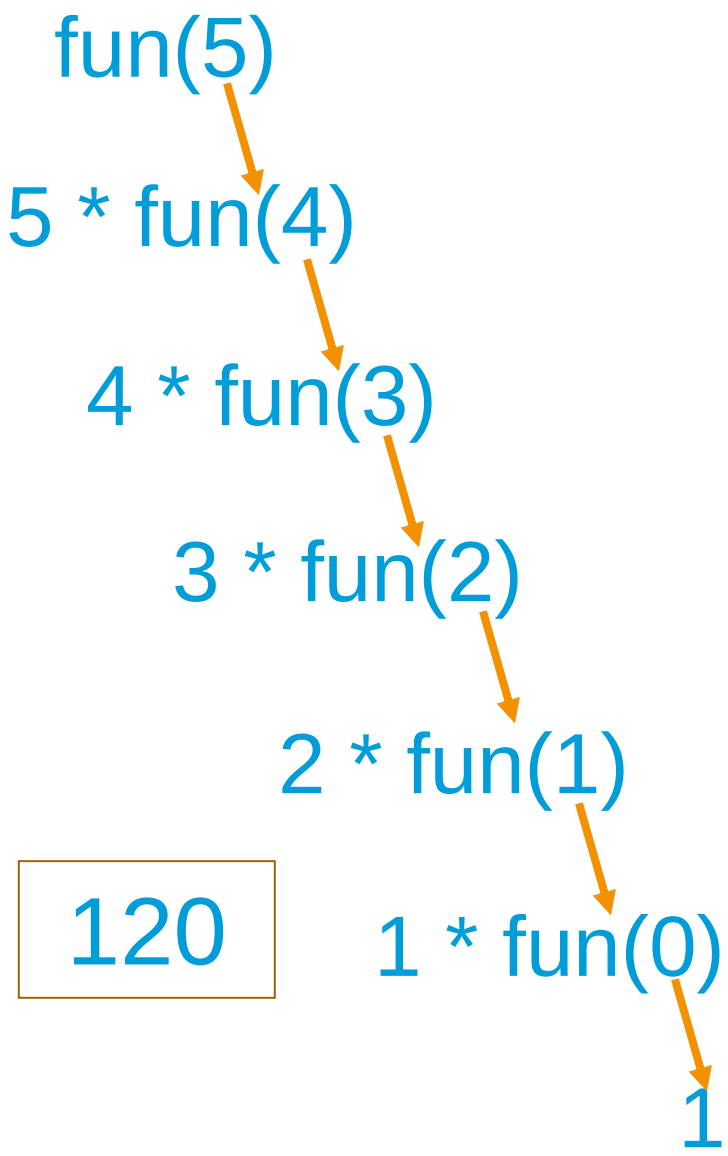
Ví dụ: Cho biết kết quả thực hiện chương trình

```
#include <stdio.h>

int fun(int a){
  a++;
  return a;
}

int main(){
  printf("%d\n", fun(fun(fun(3))));
  return 0;
}
```



Ví dụ: Cho biết kết quả thực hiện chương trình

```
#include<stdio.h>

int fun(int n){
  if(n==0) return 1;
  else return n*fun(n-1);
}
int main(){
  printf("%d\n", fun(5));
  return 0;
}
```



Ví dụ 1: Tính TBC f(a), f(b), f(c) nếu $f(x) = x^5 + \sqrt[5]{x}$

```
#include <stdio.h>
#include <math.h>
float f(float x) {
  if(x == 0.0)
    return 0;
  else
    return pow(x,5)+x/fabs(x) * pow(fabs(x), 0.2);
}
int main() {
  float a, b, c;
  printf("So 3 so thuc: "); scanf("%f%f%f", &a, &b, &c);
  printf("Ket qua: %f", (f(a) + f(b) + f(c))/3);
  return 0;
}
```

```
So 3 so thuc: 1.5 3 2.5
Ket qua: 117.260442
--------------------------------
Process exited after 9.457 seconds with return value 0
Press any key to continue . . .
```

Ví dụ 2: Tìm ƯSCLN của dãy số

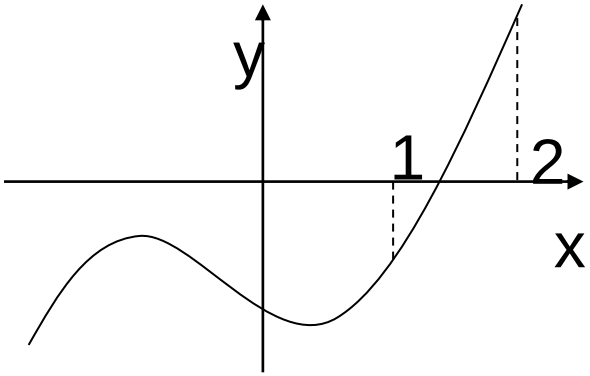
```c
#include <stdio.h>
int uscln(int a, int b) {
  a = abs(a);b = abs(b);
  while (a != b){
    if(a > b) a = a - b;
    else b = b - a;
  }
  return a;
}
int main() {
  int A[100], N, i, r;
  printf("So phan tu: "); scanf("%d", &N);
  for(i=0; i < N; i++) {
    printf("A[%d] = ", i+1); scanf("%d", &A[i]);
  }
  r = A[0];
  for(i = 1; i < N; i++)
    r = uscln(r,A[i]);
  printf("Ket qua: %d\n", r);
  return 0;
}
```

```
So phan tu: 5
A[1] = 12
A[2] = 4
A[3] = 16
A[4] = 20
A[5] = 40
Ket qua: 4

--------------------
```

Giải phương trình f(x)=0 trên đoạn [a,b]

```
#include <stdio.h>
#include <math.h>
float f(float x) {
    return x*x*x-x-1;
}
int main(){
    float a = 1.0, b = 2.0, c, eps = 1.0e-6;
    do{
        c = (a+b)/2;
        if (f(a)*f(c)<0)
            b = c;
        else
            a = c;
    }while(fabs(b-a) > eps);
    printf("Nghiem la: %.4f", (b+a)/2);
    return 0;
}
```

Giải phương trình $x^3-x-1=0$




```
Nghiem la: 1.3247
--------------------------------
Process exited after 0.1247 seconds
Press any key to continue . . .
```

VD Đọc tọa độ 3 điểm A,B,C và đưa ra diện tích ΔABC.

```
#include <stdio.h>
#include <math.h>

typedef struct{
  float x, y;
}Point;

float kc(Point A, Point B) {
  return sqrt(pow(A.x - B.x, 2) + pow(A.y - B.y, 2));
}
```

```
Toa do A(x,y): 0 1
Toa do B(x,y): 2 2
Toa do C(x,y): 2 1
Dien tich ABC: 1.0000
--------------------------------

Process exited after 10.46 seconds
Press any key to continue . . .
```

VD Đọc tọa độ 3 điểm A,B,C và đưa ra diện tích ΔABC.

```
Toa do A(x,y): 0 1
Toa do B(x,y): 2 2
Toa do C(x,y): 2 1
Dien tich ABC: 1.0000
--------------------------------
Process exited after 10.46 seconds
Press any key to continue . . .
```

```
int main(){
  Point A, B, C;
  float AB, BC, CA, p, S;
  printf("Toa do A(x,y): "); scanf("%f%f", &A.x, &A.y);
  printf("Toa do B(x,y): "); scanf("%f%f", &B.x, &B.y);
  printf("Toa do C(x,y): "); scanf("%f%f", &C.x, &C.y);
  AB = kc(A,B); BC = kc(B,C); CA = kc(C,A);
  p = (AB + BC + CA)/2;
  S = sqrt(p*(p-AB)*(p-BC)*(p-CA));
  printf("Dien tich ABC: %.4f",S);
  return 0;
}
```

Chương 5: Hàm

5.1. Khái niệm hàm
- Khái niệm chương trình con
- Phân loại: hàm và thủ tục

5.2. Khai báo và sử dụng hàm
- Khai báo và sử dụng

5.3. Phạm vi của biến
- Toàn cục và địa phương
- Biến static, biến register

5.4. Truyền tham số
- Truyền theo giá trị, truyền theo địa chỉ

Phạm vi

- Phạm vi:
  - Khối lệnh, chương trình con, chương trình chính
- Biến chỉ có tác dụng trong phạm vi được khai báo
- Trong cùng một phạm vi các biến phải có tên khác nhau.

Tình huống

- Trong hai phạm vi khác nhau có hai biến cùng tên. Trong đó một phạm vi này nằm trong phạm vi kia?

```
#include<stdio.h>
int i;
int binhphuong(int x){
  int y;
  y =  x * x;
  return y;
}
int main(){
  int y;
  for (i = 0; i <= 10; i++){
    y = binhphuong(i);
    printf("%d  ", y);
  }
  return 0;
}
```

Phân loại biến

- Biến toàn cục:
  - Biến được khai báo trong chương trình chính, được đặt sau khai báo tệp tiêu đề.
- Biến cục bộ:
  - biến được khai báo trong lệnh khối hoặc chương trình con, được đặt trước các câu lệnh.

Ghi chú

- Hàm main() cũng là một chương trình con nhưng là nơi chương trình được bắt đầu
- Biến khai báo trong hàm main() cũng là biến cục bộ, chỉ có phạm vi trong hàm main().

Biến static

- Biến cục bộ ra khỏi phạm vi thì bộ nhớ dành cho biến được giải phóng
- Yêu cầu lưu trữ giá trị của biến cục bộ một cách lâu dài => sử dụng từ khóa static.

Cú pháp:

static <kiểu_dữ_liệu> <tên_biến>;

Ví dụ→Kết quả

```
#include <stdio.h>
void fct(){
  static int count = 1;
  printf("\nDay la lan goi ham fct thu %2d", count++);
}
int main(){
  int i;
  for (i = 0; i < 10; i++)
       fct();
  return 0;
}
```

Ví dụ

```
Day la lan goi ham fct lan thu  1
Day la lan goi ham fct lan thu  2
Day la lan goi ham fct lan thu  3
Day la lan goi ham fct lan thu  4
Day la lan goi ham fct lan thu  5
Day la lan goi ham fct lan thu  6
Day la lan goi ham fct lan thu  7
Day la lan goi ham fct lan thu  8
Day la lan goi ham fct lan thu  9
Day la lan goi ham fct lan thu 10
--------------------------------
Process exited after 0.01999 seconds
Press any key to continue . . .
```

Biến register

- Thanh ghi có tốc độ truy cập nhanh hơn RAM, bộ nhớ ngoài
- Lưu biến trong thanh ghi sẽ tăng tốc độ thực hiện chương trình

Cú pháp

register <kiểu_dữ_liệu> tên_biến;

Lưu ý:

– số lượng biến register không nhiều và thường chỉ với kiểu dữ liệu nhỏ như int, char

Chương 5: Hàm

5.1. Khái niệm hàm
- Khái niệm chương trình con
- Phân loại: hàm và thủ tục

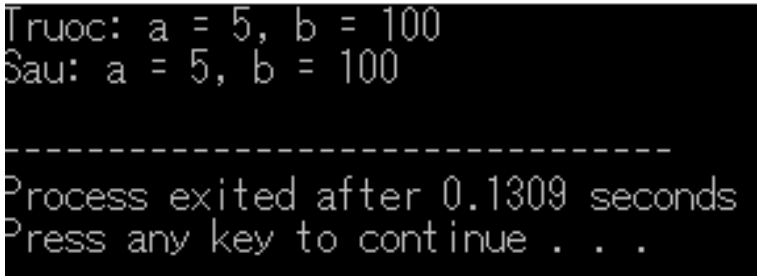
5.2. Khai báo và sử dụng hàm
- Khai báo và sử dụng

5.3. Phạm vi của biến
- Toàn cục và địa phương
- Biến static, biến register

5.4. Truyền tham số
- Truyền theo giá trị, truyền theo địa chỉ

Ví dụ

```
#include <stdio.h>
void swap(int a, int b){
  int x = a;
  a = b;
  b = x;
}
int main() {
  int a = 5, b = 100;
  printf("Truoc: a = %d, b = %d\n", a, b);
  swap(a, b);
  printf("Sau: a = %d, b = %d\n", a, b);
  return 0;
}
```



Truyền theo giá trị và truyền theo biến

- Truyền theo trị
  - Dựa trên nguyên tắc truyền những bản sao của biến được truyền
  - Những câu lệnh thay đổi giá trị tham số hình thức sẽ không ảnh hưởng tới biến được truyền
- Truyền theo biến
  - Tham số được truyền sẽ thực sự là biến và các thao tác sẽ thi hành trực tiếp với biến
  - Những câu lệnh thay đổi giá trị tham số hình thức sẽ ảnh hưởng tới biến được truyền

Truyền theo biến

- Thực chất là truyền theo địa chỉ của biến
- Khi khai báo hàm [tham số có kiểu "*địa chỉ*"]:
  - Khai báo là một con trỏ, trỏ tới một đối tượng có kiểu muốn truyền vào
  - Ví dụ: void swap (int *pa, int *pb);
- Khi truyền tham số
  - Địa chỉ của biến được truyền

    Ví dụ: swap(&a, &b)
  - Có thể truyền tên của mảng
    - Tên mảng là hằng địa chỉ

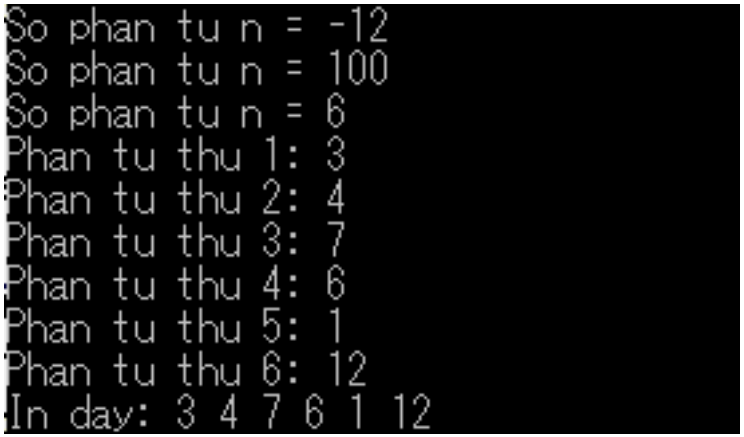
Ví dụ: Truyền theo địa chỉ của biến

```
#include <stdio.h>

void swap(int * pa, int * pb) {
  int x = *pa;
  *pa = *pb;
  *pb = x;
}

int main() {
  int a = 5, b = 100;
  printf("Truoc: a = %d, b = %d \n\n", a, b);
  swap(&a, &b);
  printf("Sau  : a = %d, b = %d \n\n", a, b);
  return 0;
}
```

Ví dụ

Nhập số nguyên dương n (100>n>0)

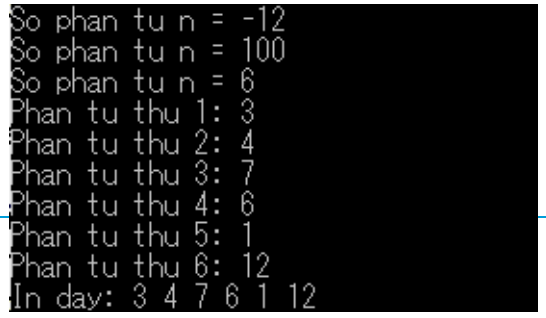
- Viết hàm nhập dãy số n số nguyên
- Viết hàm hiển thị dãy số vừa nhập trên 1 dòng, mỗi số cách nhau 1 dấu cách ‘ ’.



Ví dụ

```
#include<stdio.h>
#include<math.h>

int n, a[10000]; //bien toan cuc

void nhap(){
   int i;
   do{
        printf("So phan tu n = ");
        scanf("%d", &n);
   }while(n<=0 ||n>= 100);

   for(i = 0;i<n;i++){
        printf("Phan tu thu %d: ",i+1);
        scanf("%d", &a[i]);
   }
}
```

```
void xuat(){
   int i;
   printf("In day:");
   for(i = 0;i<n;i++)
    printf(" %d",a[i]);
}

int main() {
   nhap();
   xuat();
   return 0;
}
```



Ví dụ

```
#include<stdio.h>
#include<math.h>

void nhap(int a[1000], int n){
   int i;
   for(i=0;i<n;i++){
        printf("Phan tu thu %d: ",i+1);
        scanf("%d", &a[i]);
   }
}
void xuat(const int a[1000], int n){
   int i;
   printf("In day:");
   for(i=0;i<n;i++)
        printf(" %d",a[i]);
}
```

```
int main() {
   int n, a[1000];
   do{
    printf("So phan tu n = ");
    scanf("%d", &n);
   }while(n<=0 ||n>= 100);
   nhap(a,n);
   xuat(a,n);
   return 0;
}
```

Ví dụ

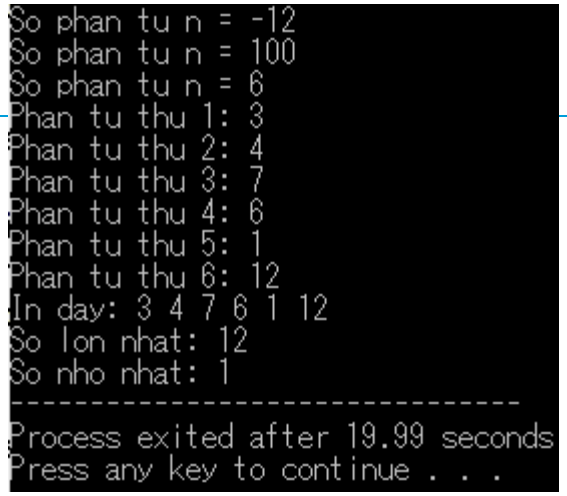
```
#include<stdio.h>
#include<math.h>
void nhap(int a[100], int *pn){
   int i;
   int n;
   do{
    printf("So phan tu n = ");
   scanf("%d", &n);
   }while(n<=0 || n>= 100);
   *pn = n;
   for(i = 0;i<n;i++){
        printf("Phan tu thu %d: ",i+1);
    scanf("%d", &a[i]);
   }
}
```

```
void xuat(int a[], int n){
   int i;
   printf("In day:");
   for(i = 0;i<n;i++)
    printf(" %d",a[i]);
}
int main() {
   int N, A[100];
   nhap(A,&N);
   xuat(A,N);
   return 0;
}
```

Ví dụ vận dụng 1

Nhập số nguyên dương n (100>n>0)

- Viết hàm nhập dãy số n số nguyên
- Viết hàm hiển thị dãy số vừa nhập trên 1 dòng, mỗi số cách nhau 1 dấu cách ‘ ’.
- Viết hàm tìm min/max



Ví dụ vận dụng 1

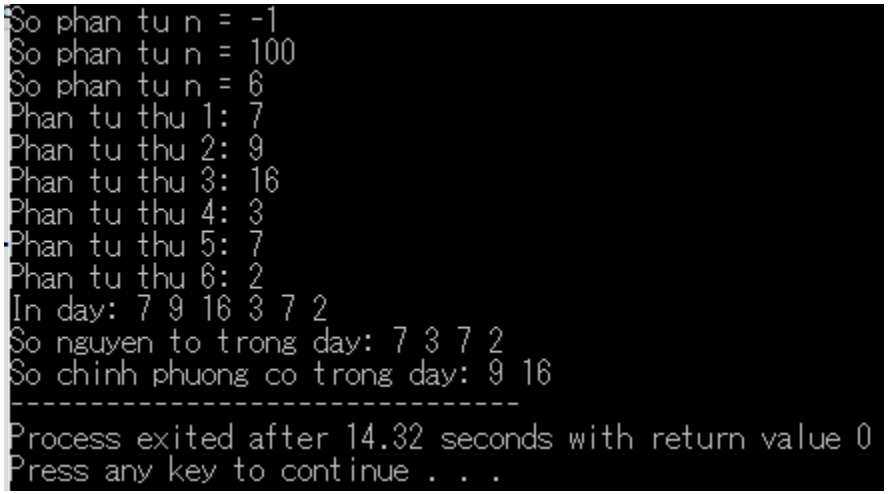
```
#include<stdio.h>
#include<math.h>

void nhap(int a[100], int n){
	int i;
	for(i=0;i<n;i++){
	printf("Phan tu thu %d: ",i+1);
	  scanf("%d", &a[i]);
	}
}
void xuat(int a[], int n){
	int i;
	printf("In day:");
	for(i=0;i<n;i++)
	  printf(" %d",a[i]);
}
int tim_max(int b[], int n){
	int i, max = b[0];
	for(i=1;i<n;i++)
	  if(max<b[i])
		max = b[i];
	return max;
}
```

```
int tim_min(int b[], int n){
	int i, min = b[0];
	for(i=1;i<n;i++)
	  if(min>b[i])
		min = b[i];
	return min;
}
int main() {
	int n, a[100];
	do{
	  printf("So phan tu n = "); scanf("%d", &n);
	}while(n<=0 ||n>= 100);
	nhap(a,n);
	xuat(a,n);
	printf("\nSo lon nhat: %d", tim_max(a,n));
	printf("\nSo nho nhat: %d",tim_min(a,n));
	return 0;
}
```

Ví dụ vận dụng 2

Nhập số nguyên dương n (100>n>0)

- Viết hàm nhập dãy số n số nguyên
- Viết hàm hiển thị dãy số vừa nhập trên 1 dòng, mỗi số cách nhau 1 dấu cách ‘ ’.
- Viết hàm kiểm tra số nguyên n có là số nguyên tố?
- Viết hàm liệt kê các số nguyên tố có trong dãy số đã nhập
- Viết hàm kiểm tra số nguyên n có là chính phương?
- Viết hàm in các số chính phương có trong dãy.



Ví dụ vận dụng 2

```
#include<stdio.h>
#include<math.h>
void nhap(int b[], int n){
        int i;
        for(i=0;i<n;i++){
          printf("Phan tu thu %d: ",i+1);
          scanf("%d", &b[i]);
        }
}
void xuat(int b[], int n){
        int i;
        printf("In day:");
        for(i=0;i<n;i++)
          printf(" %d",b[i]);
}
int nguyento(int n) {
        int i;
        if (n <= 1)
          return 0;
        for (i = 2; i <= sqrt(n); i++)
          if (n%i == 0)
                  return 0;
        return 1;
}
int chinhphuong(int n) {
        if (n>=0 && sqrt(n) == floor(sqrt(n)))
          return 1;
        else   return 0;
}
```

```
void InNT(int a[], int n){
        int i;
        printf("\nSo nguyen to trong day:");
        for(i=0;i<n;i++)
          if(nguyento(a[i]))
                  printf(" %d",a[i]);
}
void InCP(int a[], int n){
        printf("\nSo chinh phuong co trong day:");
        int i;
        for(i=0;i<n;i++)
          if(chinhphuong(a[i]))
                  printf(" %d",a[i]);
}
int main() {
        int a[100], n;
        do{
          printf("So phan tu n = "); scanf("%d", &n);
        }while(n<=0 ||n>= 100);
        nhap(a,n);
        xuat(a,n);
        InNT(a,n);
        InCP(a,n);
        return 0;
}
```

Ví dụ vận dụng 2

```
#include<stdio.h>
#include<math.h>
void nhap(int a[], int *pn){
        int i;
        int n;
        do{
          printf("So phan tu n = ");
          scanf("%d", &n);
        }while(n<=0 || n>= 100);
        *pn = n;
        for(i = 0;i<n;i++){
          printf("Phan tu thu %d: ",i+1);
          scanf("%d", &a[i]);
        }
}
void xuat(int a[], int n){
        int i;
        printf("In day:");
        for(i=0;i<n;i++)
          printf(" %d",a[i]);
}
int nguyento(int n) {
        int i;
        if (n <= 1)   return 0;
        for (i = 2; i <= sqrt(n); i++)
          if (n%i == 0)
                  return 0;
        return 1;
}
```

```
int chinhphuong(int n) {
        if (n>=0 && sqrt(n) == floor(sqrt(n)))
          return 1;
        else   return 0;
}
void InNT(int a[], int n){
        int i;
        printf("\nSo nguyen to trong day:");
        for(i=0;i<n;i++)
          if(nguyento(a[i]))
                  printf(" %d",a[i]);
}
void InCP(int a[], int n){
        printf("\nSo chinh phuong co trong day:");
        int i;
        for(i=0;i<n;i++)
          if(chinhphuong(a[i]))
                  printf(" %d",a[i]);
}
int main() {
        int A[100], N;
        nhap(A,&N);
        xuat(A,N);
        InNT(A,N);
        InCP(A,N);
        return 0;
}
```