

HÌNH CẦU

- Nguyễn Hoàng Yến Như
- Nguyễn Trần Phúc Nghi
- Nguyễn Trần Phúc An
- Nguyễn Đức Anh Phúc
- Trịnh Thị Thanh Trúc
- KS. Hồ Thái Ngọc
- KS. Cao Bá Kiệt
- KS. Lê Ngọc Huy
- CN. Bùi Cao Doanh
- CN. Nguyễn Trọng Thuận
- KS. Phan Vĩnh Long
- KS. Nguyễn Cường Phát
- ThS. Nguyễn Hoàng Ngân
- ThS. Đỗ Văn Tiến
- ThS. Nguyễn Hoàn Mỹ
- ThS. Dương Phi Long
- ThS. Trương Quốc Dũng
- ThS. Nguyễn Thành Hiệp
- ThS. Nguyễn Võ Đăng Khoa
- ThS. Võ Duy Nguyên
- ThS. Trần Việt Thu Phương
- TS. Nguyễn Tấn Trần Minh Khang

Khai báo kiểu dữ liệu

```
101.struct DiemKhongGian
102.{
103.    float x;
104.    float y;
105.    float z;
106.};
107.typedef struct DiemKhongGian DIEMKHONGGIAN;
```

Khai báo kiểu dữ liệu

```
101.struct HinhCau  
102.{  
103.|    DIEMKHONGGIAN I;  
104.|    float R;  
105.};  
106.typedef struct HinhCau HINHCAU;
```

Nhập hình cầu

— Định nghĩa hàm

```
101. void Nhap(DIEMKHONGGIAN &P)
102. {
103.     cout << "Nhap x: ";
104.     cin >> P.x;
105.     cout << "Nhap y: ";
106.     cin >> P.y;
107.     cout << "Nhap z: ";
108.     cin >> P.z;
109. }
```

Nhập hình cầu

— Định nghĩa hàm

```
101. void Nhap(HINHCAU &c)
102. {
103.     cout << "Nhap tam: ";
104.     Nhap(c.I);
105.     cout << "Nhap ban kinh: ";
106.     cin >> c.R;
107. }
```

Xuất hình cầu

```
101. void Xuat(DIEMKHONGGIAN P)
102. {
103.     cout << "\n x= " << P.x;
104.     cout << "\n y= " << P.y;
105.     cout << "\n z= " << P.z;
106. }
```

Xuất hình cầu

```
101. void Xuat(HINHCAU c)
102. {
103.     cout << "\n Tam: ";
104.     Xuat(c.I);
105.     cout << "\n Ban kinh:" << c.R;
106. }
```

Tính diện tích xung quanh

— Định nghĩa hàm

```
101.float DienTich(HINHCAU c)
102.{
103.|    return 4 * 3.14 * c.R * c.R;
104.}
```

$$S = 4\pi R^2$$

Tính thể tích

— Định nghĩa hàm

```
101.float TheTich(HINHCAU c)
102.{
103.|    return 4 * 3.14 * c.R * c.R * c.R / 3;
104.}
```

$$V = \frac{4}{3} \pi R^3$$

Kiểm tra một điểm thuộc hình cầu

```
101.float  KhoảngCach(DIEMKHONGGIAN P,DIEMKHONGGIAN Q)
102.{
103.    return sqrt((P.x-Q.x)*(P.x-Q.x)
104.                +(P.y-Q.y)*(P.y-Q.y)
105.                +(P.z-Q.z)*(P.z-Q.z));
106.}
```

Kiểm tra một điểm thuộc hình cầu

```
101.int KtThuoc(HINHCAU c,DIEMKHONGGIAN P)
102.{
103.|    float kc = KhoangCach(c.I,P);
104.|    if(kc<=c.R)
105.|        return 1;
106.|    return 0;
107.}
```

Vị trí tương đối của hai hình cầu

— Các trường hợp:

+ Trùng nhau 

+ Rời nhau 

+ Tiếp xúc ngoài 

+ Cắt nhau 

+ Tiếp xúc trong 

+ Chứa trong nhau 

— Hàm trả về một trong 6 giá trị

+ 0. Trùng nhau 

+ 1. Rời nhau 

+ 2. Tiếp xúc ngoài 

+ 3. Cắt nhau 

+ 4. Tiếp xúc trong 

+ 5. Chứa trong nhau 

Xét vị trí tương đối của hai hình cầu

— Định nghĩa hàm

```
101.float  KhoangCach(DIEMKHONGGIAN P,DIEMKHONGGIAN Q)
102.{
103.    |    return sqrt((P.x-Q.x)*(P.x-Q.x)
104.    |        +(P.y-Q.y)*(P.y-Q.y)
105.    |        +(P.z-Q.z)*(P.z-Q.z));
106.}
```

Xét vị trí tương đối của hai hình cầu


```

101.int  TuongDoi(HINHCAU C1,HINHCAU C2)
102.{
103.    float kc = KhoangCach(c1.I,c2.I)
104.    if (kc==0 && c1.R==c2.R)
105.        return 0;
106.    if(kc>c1.R+c2.R)
107.        return 1;
108.    if(kc==c1.R+c2.R)
109.        return 2;
110.    ...
111.}
    
```

– Hàm trả về một trong 6 giá trị

+ 0. Trùng nhau 

+ 1. Rời nhau 

+ 2. Tiếp xúc ngoài 

+ 3. Cắt nhau 

+ 4. Tiếp xúc trong 

+ 5. Chứa trong nhau 

Xét vị trí tương đối của hai hình cầu

```

101.int  TuongDoi(HINHCAU C1,HINHCAU C2)
102.{
103.    ...
104.    if(kc<c1.R+c2.R && kc>abs(c1.R-
105.        return 3;
106.    if(kc==abs(c1.R-c2.R))
107.        return 4;
108.    return 5;
109.}

```

— Hàm trả về một trong 6 giá trị

+ 0. Trùng nhau 

+ 1. Rời nhau 

+ 2. Tiếp xúc ngoài 

+ 3. Cắt nhau 

+ 4. Tiếp xúc trong 

+ 5. Chứa trong nhau 

Xét vị trí tương đối của hai hình cầu


```

101.int TuongDoi(HINHCAU C1,HINHCAU C2)
102.{
103.    float kc = KhoangCach(c1.I,c2.I);
104.    if (kc==0 && c1.R==c2.R)
105.        return 0;
106.    if(kc>c1.R+c2.R)
107.        return 1;
108.    if(kc==c1.R+c2.R)
109.        return 2;
110.    if(kc<c1.R+c2.R && kc>abs(c1.R-c2.R))
111.        return 3;
112.    if(kc==abs(c1.R-c2.R))
113.        return 4;
114.    return 5;
115.}
    
```

– Hàm trả về một trong 6 giá trị

+ 0. Trùng nhau 

+ 1. Rời nhau 

+ 2. Tiếp xúc ngoài 

+ 3. Cắt nhau 

+ 4. Tiếp xúc trong 

+ 5. Chứa trong nhau 

Chúc các bạn học tốt
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN TP.HCM

Nhóm UIT-Together
Nguyễn Tấn Trần Minh Khang