Phương pháp Lập trình Hướng đối tượng

Các Kiểu Lập trình

GV: Lê Xuân Định

Các Kiểu Lập trình
(Programming Paradigms)

- Lập trình **Mệnh lệnh** (Imperative Programming)
  - Ngôn ngữ LT: Hợp ngữ, C đơn giản, …
  - Đơn vị của chương trình là **lệnh**.
- Lập trình **Thủ tục** (Procedural Programming)
  - Ngôn ngữ LT: C, Pascal, …
  - Đơn vị của chương trình là **thủ tục / hàm / ctrình con**.
- Lập trình **Hướng đối tượng** (Object Oriented Programming)
  - Ngôn ngữ LT: C++, Java, C#, …
  - Đơn vị của chương trình là **đối tượng / lớp**.
- Và nhiều kiểu lập trình khác: LTr Khai báo, LTr Hàm, LTr Logic, LTr Hướng sự kiện, LTr Hướng dịch vụ, v.v.

Chú ý: Cách phân chia ra thành các "kiểu lập trình" như thế này chỉ mang tính tương đối.

Lập trình Mệnh lệnh

- Chương trình là một danh sách các **câu lệnh**.
- Ví dụ: Chương trình "Vẽ hình vuông"

```
drawRight(100);
drawDown(100);
drawLeft(100);
drawUp(100);
```

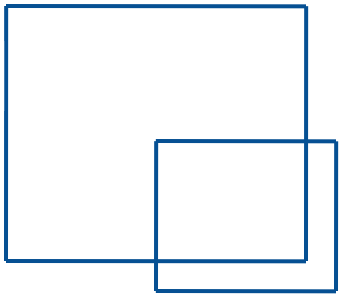
Lập trình Mệnh lệnh

- Chương trình là một danh sách các **câu lệnh**.
- Ví dụ 2: Chương trình "Vẽ hai hình vuông"

```
movePen(0, 0);
drawRight(100);
drawDown(100);
drawLeft(100);
drawUp(100);
movePen(50, 50);
drawRight(60);
drawDown(60);
drawLeft(60);
drawUp(60);
```

Dài!

Lặp code

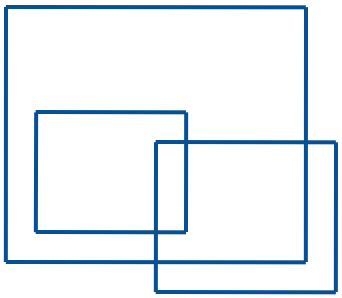


Lập trình Thủ tục

- Chương trình được chia ra thành nhiều **chương trình con** (thủ tục, hàm).
  - Mỗi chương trình con là một danh sách các câu lệnh.
  - Chương trình con này có thể ***gọi*** *ctrình con* khác.
- Ví dụ: Chương trình "Vẽ ba hình vuông"

```
void vuong(int w,
     int x, int y)
{ movePen(x, y);
  drawRight(w);
  drawDown(w);
  drawLeft(w);
  drawUp(w);
}
```

```
void main(){
  vuong(100, 0, 0);
  vuong(60, 50, 50);
  vuong(50, 10, 40);
}
```



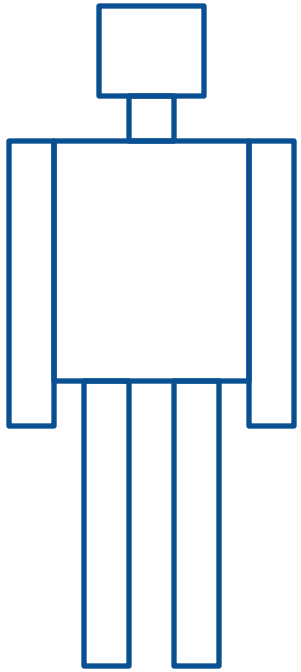
Lập trình Thủ tục

- Chương trình được chia ra thành nhiều **chương trình con** (thủ tục, hàm).
- Ví dụ 2: Chương trình "Vẽ robot (vuông & chữ nhật)"

```
void vuong(int w,
     int x, int y)
{ movePen(x, y);
  drawRight(w);
  ...
}
```

```
void chuNhat(
     int w, int h,
     int x, int y)
{ movePen(x, y);
  drawRight(w);
  drawDown(h);
  ...
}
```

```
void main(){
  color(8, 80, 145);
  vuong(20, -10,0);
  vuong(10, -5,20);
  chuNhat(40,50, -20,30);
  chuNhat(10,60, -30,30);
  chuNhat(10,60,  20,30);
  chuNhat(10,60, -15,80);
  chuNhat(10,60,   5,80);
}
```



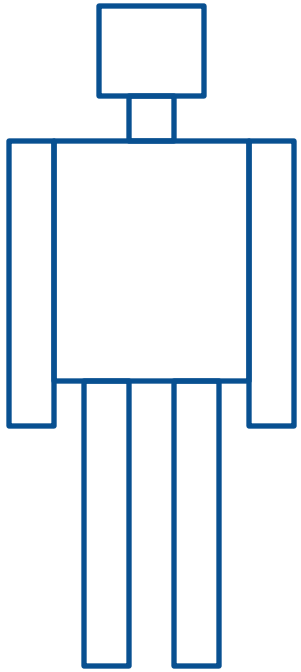
Lập trình Thủ tục

- Chương trình được chia ra thành nhiều **chương trình con** (thủ tục, hàm).
- Ví dụ 3: Chương trình "Vẽ robot nhảy"

```
void robot(int w,
 int r, int g, int b,
 int x, int y){
  color(r, g, b);
  vuong(2*w, x-w, y);
  vuong(w, x-w/2, y+2*w);
  chuNhat(4*w,5*w, x-2*w, y+3*w);
  chuNhat(w,6*w,   x-3*w, y+3*w);
  chuNhat(w,6*w,   x+2*w, y+3*w);
  chuNhat(w,6*w, x-3*w/2,y+8*w);
  chuNhat(w,6*w,   x+w/2,y+8*w);
}
```

```
void xoaRobot(int w,
        int x, int y){
  robot(w, 0,0,0, x,y);
}
```

```
void main()
{ for(int i=0;;i++)
  {sleep(200);
   xoaRobot(10, 0, 10*(i%2))
   robot(10, 8,80,145,
         0, 10*((i+1)%2) );
} }
```



Lập trình Thủ tục

- Chương trình được chia ra thành nhiều **chương trình con** (thủ tục, hàm).
- Ví dụ 3: Chương trình "Vẽ robot nhảy"

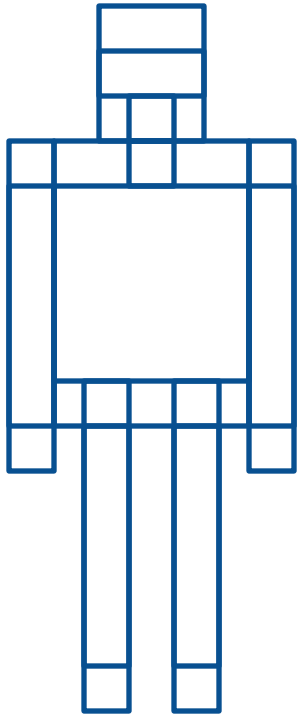
```
void robot(int w,
 int r, int g, int b,
 int x, int y){
  color(r, g, b);
  vuong(2*w, x-w, y);
  vuong(w, x-w/2, y+2*w);
  chuNhat(4*w,5*w, x-2*w, y+3*w);
  chuNhat(w,6*w,   x-3*w, y+3*w);
  chuNhat(w,6*w,   x+2*w, y+3*w);
  chuNhat(w,6*w, x-3*w/2,y+8*w);
  chuNhat(w,6*w,   x+w/2,y+8*w);
}
```

Dài!

```
void xoaRobot(int w,
        int x, int y){
  robot(w, 0,0,0, x,y);
}
```

```
void main()
{ for(int i=0;;i++)
  {sleep(200);
   xoaRobot(10, 0, 10*(i%2))
   robot(10, 8,80,145,
         0, 10*((i+1)%2) );
} }
```

Dài!



Lập trình Thủ tục với Struct

- Chương trình được chia ra thành nhiều **chương trình con** cùng các **cấu trúc dữ liệu (struct)**.
- Ví dụ: Chương trình "Vẽ robot nhảy"

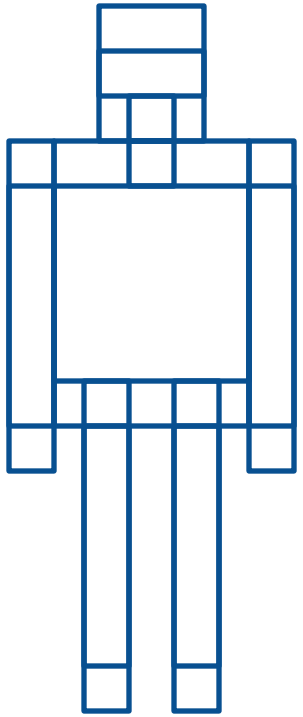
```
struct Robot{ int w,
  int r, int g, int b,
  int x, int y  };
```

```
void veRobot(Robot r){
  color(r.r, r.g, r.b);
  vuong(2*r.w, r.x-r.w, r.y);
  ...
}
```

```
void doiRobot(Robot r,
              int x, int y)
{ r.x = x;  r.y = y; }
```

```
void xoaRobot(Robot r){
  Robot rx = {r.w, 0,0,0, r.x, r.y};
  veRobot(rx);
}
```

```
void main()
{ Robot r={10, 8,80,145, 0,0};
  for(int i=0;;i++)
  { sleep(200); xoaRobot(r);
    doiRobot(r, 0, 10*(i%2));
    veRobot(r);
} }
```

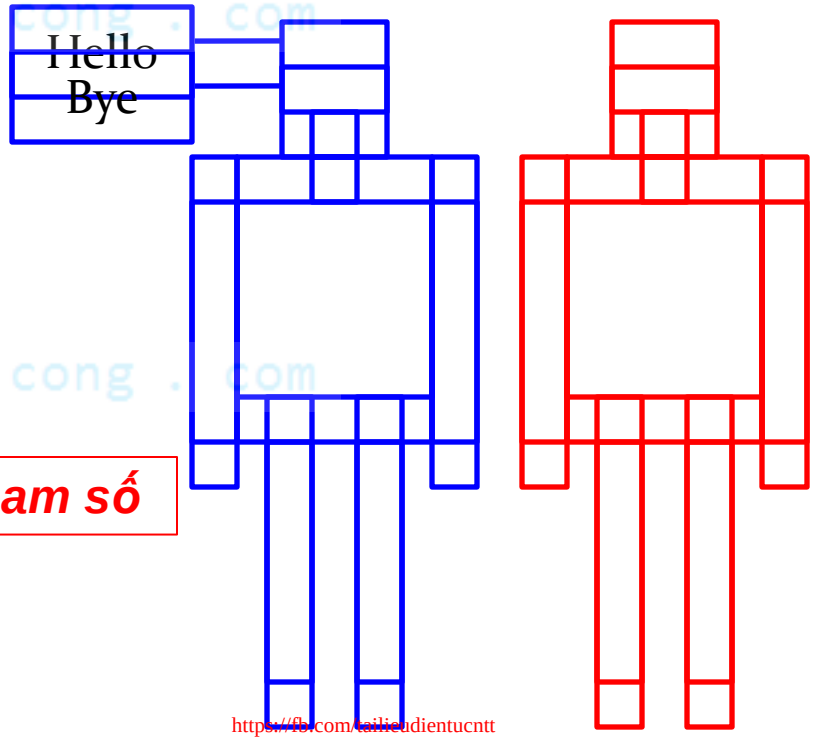


Lập trình Thủ tục với Struct

- Chương trình được chia ra thành nhiều **chương trình con** cùng các **cấu trúc dữ liệu (struct)**.
- Ví dụ 2: Chương trình "Vẽ robot nhảy & robot chào"

```
...
void main()
{ Robot r={10, 255,0,0, 0,0};
  RobotChao rc={10,0,0,255,
                -70,0, 'E'};
  helloRobot(rc);
  nhayRobot(rc, 9);
  byeRobot(rc);
  nhayRobot(r, 100);
}
```

Khác kiểu tham số



Lập trình Thủ tục với Struct

- Chương trình được chia ra thành nhiều **chương trình con** cùng các **cấu trúc dữ liệu (struct)**.
- Ví dụ 2: Chương trình "Vẽ robot nhảy & robot chào"

```
struct Robot{ int w,
  int r, int g, int b,
  int x, int y  };
```

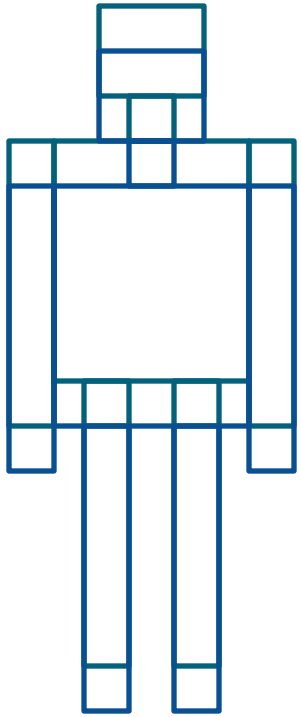
```
void veRobot(Robot r){
  color(r.r, r.g, r.b);
  vuong(2*r.w, r.x-r.w, r.y);
  ...
}
```

```
void xoaRobot(Robot r){
  Robot rx = {r.w, 0,0,0,
              r.x, r.y};
  veRobot(rx); }
```

```
struct RobotChao{ int w,
  int r, int g, int b,
  int x, int y, char nn };
```

```
void veRobot(RobotChao r){
  color(r.r, r.g, r.b);
  vuong(2*r.w, r.x-r.w, r.y);
  ...
}
```

```
void xoaRobot(RobotChao r){
  Robot rx = {r.w, 0,0,0,
              r.x, r.y};
  veRobot(rx); }
```

Lập trình Thủ tục với Struct

- Ví dụ 2: Chương trình "Vẽ robot nhảy & robot chào"
- Thảo luận: Làm thế nào ***tái sử dụng*** *struct Robot và các hàm* veRobot(), nhayRobot(),... cho struct RobotChao?

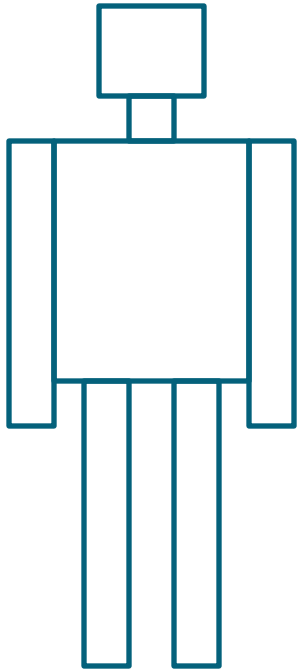
```
struct Robot{ int w,
  int r, int g, int b,
  int x, int y  };
```

```
void veRobot(Robot r);
void xoaRobot(Robot r);
void doiRobot(Robot r);
void nhayRobot(Robot r);
```

```
struct RobotChao{ int w,
  int r, int g, int b,
  int x, int y, char nn };
```

```
void veRobot(RobotChao r);
void xoaRobot(RobotChao r);
void doiRobot(RobotChao r);
void nhayRobot(RobotChao r);
```

```
void helloRobot(RobotChao r);
void byeRobot(RobotChao r);
```



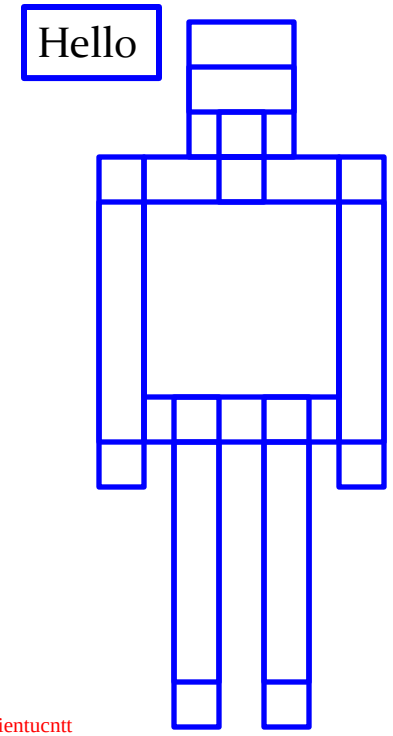
Lập trình Hướng đối tượng

- Chương trình là cuộc hội thoại giữa các **đối tượng** (đối tượng = struct + hàm).
- Ví dụ: Chương trình "Vẽ robot nhảy & robot chào"

```
class Robot{ int w,
  int r, int g, int b,
  int x, int y;
public:
  void ve();
  void xoa();
  void doi(int x, int y);
  void nhay();
};
```

```
class RobotChao
  :public Robot {
  char nn;
public:
  void hello();
  void bye();
};
```

```
void main()
{ RobotChao rc;
 rc.hello();
 rc.nhay(100);
}
```



Kết luận

- Qua mỗi bước phát triển, kiểu lập trình sau ***tích hợp nhiều đơn vị*** của kiểu lập trình trước vào một đơn vị.
  - Thủ thục là tập hợp các lệnh.
  - Đối tượng là sự tích hợp của dữ liệu và thủ tục.

→Thuận tiện cho việc *phát triển* ***ứng dụng lớn*** *hơn*!
  - Nâng cao tính tái sử dụng (reusability).
  - Nâng cao khả năng mở rộng (scalability).