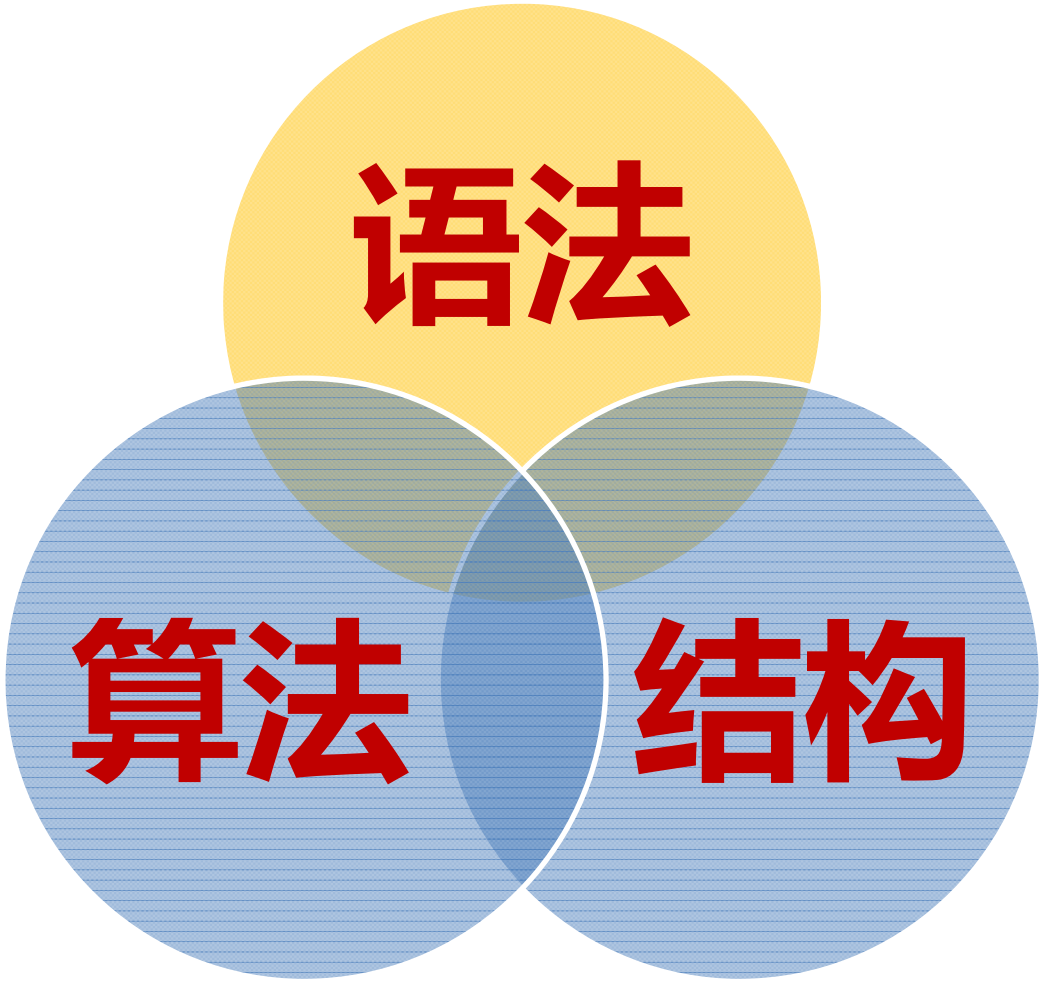
计算机科学基础I---总复习

- 进制数转换、补码及其计算
- C++基础知识（ch1）
- 基本控制结构程序设计（ch2）
- 函数（ch3）
- 数组与指针（ch5, ch5.4除外）
- 算法（ch6.2）
- 引用（ch4.4.1）

二进制

- 不同进位计数制之间的转换(掌握)
  - 十进制数与二进制、八进制、十六进制数之间的转换
  - 二进制、八进制、十六进制数之间的转换
- 机器数的表示
  - 符号表示：补码的表示及其计算(掌握)
  - 小数点表示：浮点表示(理解)

总结



数据类型

- **基本数据类型**
  - **int，char，float，double， bool， void**
- **非基本数据类型**
  - **数组，指针，引用，枚举**

TIPS：

- **不同数据类型的变量参与运算时，数据类型转换**
- **数据类型占用字节数**
- **字符数组与字符串**
- **指针与数组的关系**

基础知识—— 操作数

- **变量**
  - 标识符
  - **数据类型**
    - **关键字、字节数、取值范围**
- **常量**
  - 数字常量、文字常量
    - **转义符**
  - 常变量
    - **const**

int
char
float
double
bool
数组
——字符数组
指针
引用

一级指针与一维数组 1/2

1, 指针的定义:　指针变量的值为地址

指针必须指向某变量后引用　　int i, *p=&i;

2, 指针初始化、赋值

```
int a[10],*p;
p=a;
p=&a[0];
```

```
char a[10]="nanjing";
char *p="nanjing";
p="shanghai";
a="shanghai";  //错误
```

一级指针与一维数组 2/2

4，指针运算

```
int a[10]={1,2,3,4,5,6,7,8,9,10};
int*p,*q;
p=a;
q=&a[9];
int n=q-p;
cout<<a<<endl;
cout<<p<<‘\t’<<q<<‘\t’<<n<<endl;
cout<<*p++<<endl;
cout<<(*p)++<<endl;
cout<<++*p<<endl;
cout<<*++p<<endl;
cout<<p<<endl;
cout<<q<<endl;
```

```
0012FF20
0012FF20        0012FF44        9
1
2
4
3
0012FF28
0012FF44
```

二级指针与二维数组

- **二级指针**

  int **p1;

- **数组指针**

  int *p2[4];

- **指针数组**

  int (*p3)[4];

- **二维数组**

  int a[4][4];

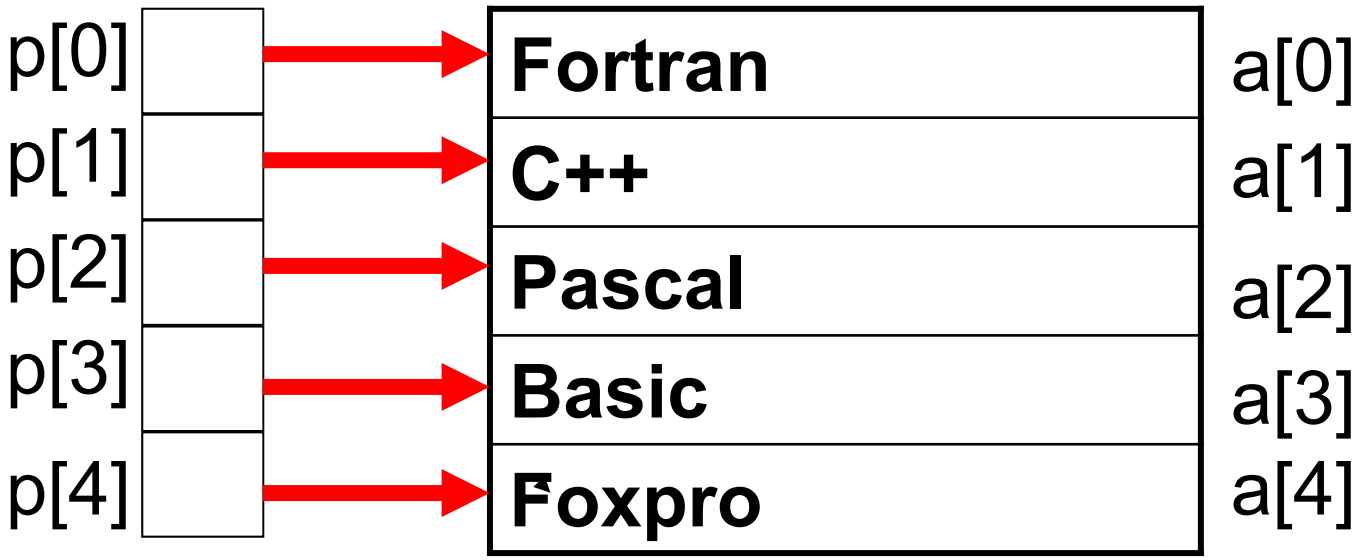
```
for(int i=0;i<4;i++)
    for(int j=0;j<4;j++)
        a[i][j]=i+j;

p3=a;

for(i=0;i<4;i++)
    p2[i]=&a[i][0];

p1=p2;
```

```
int main(){
    char a[5][80], *p[5], **q, **max;
    int i;
    for(i=0;i<5;i++)  //p[i]指向a的第i行
        p[i]=a[i];              //a[i]是第i行的首地址
    for(i=0;i<5;i++)
        cin.getline(p[i],80);
        //输入5个字符串存入字符数组a
```



运算符

- 算术运算符
- 关系运算符
- 逻辑运算符
- 自增自减运算符
- 位运算符。。。

TIPS

- 混合运算时，先判断优先级，当优先级相同时，判断结合性；

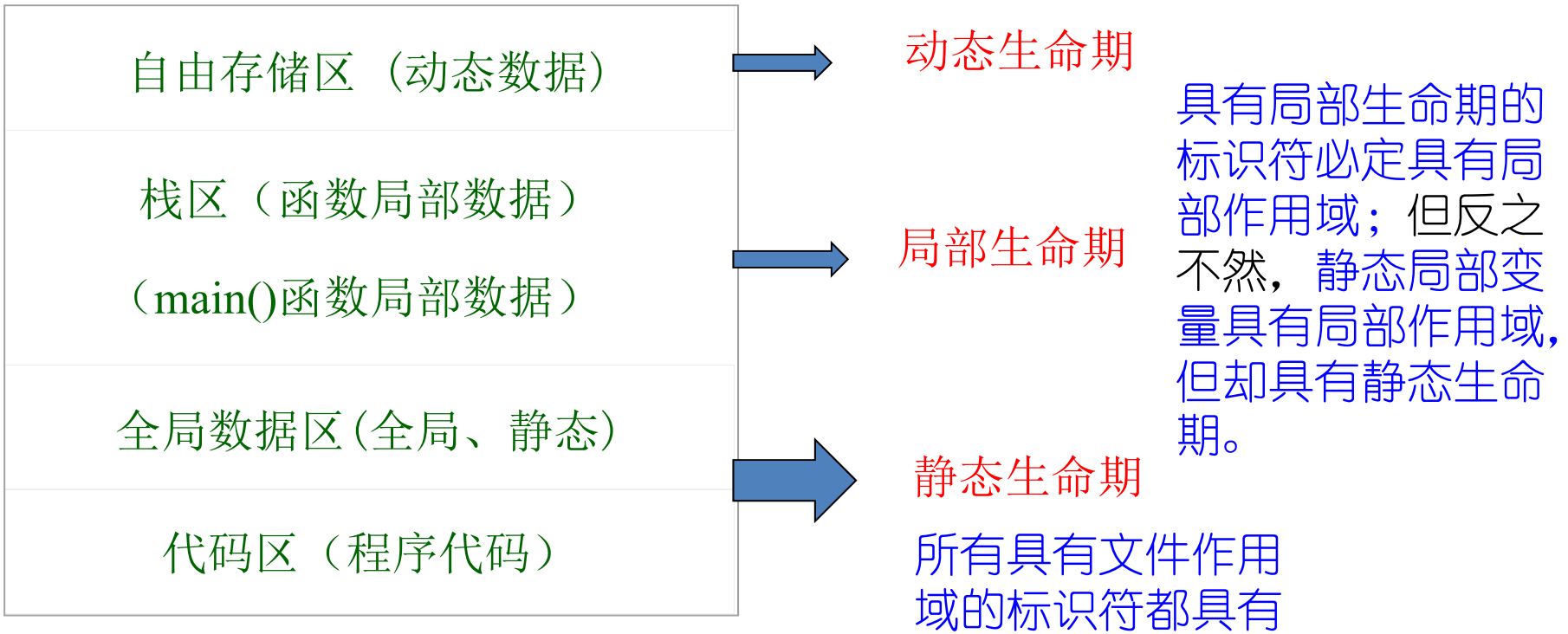
输入输出

cin, cout, cin.get(), cin.getline(), setw()， sizeof(), 格式控制

变量与常量

- 常量----字符串
- 常变量---- const
- 变量
  - 存储机制，生命期，作用域，

生命期


自由存储区（动态数据）
栈区（函数局部数据）
（main()函数局部数据）
全局数据区(全局、静态)
代码区（程序代码）
动态生命期
局部生命期
静态生命期
具有局部生命期的标识符必定具有局部作用域；但反之不然，静态局部变量具有局部作用域，但却具有静态生命期。
所有具有文件作用域的标识符都具有静态生命期。

标识符的作用域、存储机制、生命期

- 标识符
  - 变量名
    - 局部变量、局部静态变量、（外部）全局变量、全局静态变量
  - 函数名
    - （外部）函数、静态函数
- 作用域、存储机制、生命期的 区别与联系
  - 作用域——使用范围
  - 存储区域——存储类型决定存储区域，存储区域决定生命期
- 作用域
  - 块域（函数域）
  - 函数声明域
  - 文件域

- 存储机制
  - 存储类型
    - auto、register、static、extern
  - 存储区域
    - 代码区、全局数据区、栈区、自由存储区
  - 生命期
    - 静态生命期、局部生命期、动态生命期

语句

- 分支语句
- 循环语句
- 开关语句
- 转向语句

if语句基本格式：

1、if (表达式) 语句1;

if语句基本格式：

2、if (表达式) 语句1;

else 语句2;

三元运算符语法格式：

表达式1？表达式2：表达式3

开关语句(switch语句) 用来实现多选一:

```
switch (表达式) {
   case  常量表达式１:  《语句序列１》《break;》
   ……
   case 常量表达式n:《语句序列n》《break;》
    《default:语句序列》
}
```

嵌套有两种形式，嵌套在else分支中:

```
if (表达式1) 语句1;
else if (表达式2) 语句2;
     else if …
          else 语句n;
```

if和else“就近配对”

while语句基本格式（当型循环）：

while (表达式)

循环体语句

do-while语句基本格式（直到型循环）：

do 循环体语句

while(表达式)；

for循环语句的格式：

for (表达式1; 表达式2;表达式3)

循环体语句

break语句：

break语句只能用在switch语句和循环语句中，只能终止其所在的循环语句。

continue语句：

continue语句只能用在循环语句中，用来终止本次循环。

return 语句

函数

- 函数定义
  - 有参,无参,有返回值,无返回值
- 函数原型
- 函数调用
  - 传值调用
  - 传址调用
  - 引用调用
- 函数重载、默认参数的函数、内联函数

函数调用

- 函数的定义

  《数据类型》函数名（参数类型1　形式参数1《,参数类型2 形式参数2,•••》{函数体}

- 函数的调用
  - 函数声明

    《函数返回值类型》函数名（《形参表》）;
  - 参数

    传值调用——实参传递给形参
  - 返回值

    return 表达式;

- 函数的调用机制

  保护现场→分配栈空间→函数执行结束，恢复现场

传值调用

```
float power(float x,int n){
//求x的n次幂
float p=1;
while(n--)  p*=x;
return p; }
```

引用调用

```
void swap(double & d1, double & d2)
{
    double temp ;
    temp=d1 ;
    d1=d2 ;
    d2=temp ;
 }
```

传址调用

实参中的数组地址传到形参中，实参形参共用同一段内存。

```
void  swap(int a[2])
{  int t=a[0];a[0]=a[1];
      a[1]=t;
}

void exchang(int *p1，int *p2)   //指针作形参
{   int p;
    p=*p1；*p1=*p2；*p2=p;
    //p1、p2所指向的实参变量交换数据
}

void inverse(int matrix1[3][6],int middle[6][3]){ //转置矩阵
   int i,j;
   for (i=0;i<3;i++)
     for (j=0;j<6;j++)
       middle[j][i]=matrix1[i][j];
   return;}
```

常用库函数

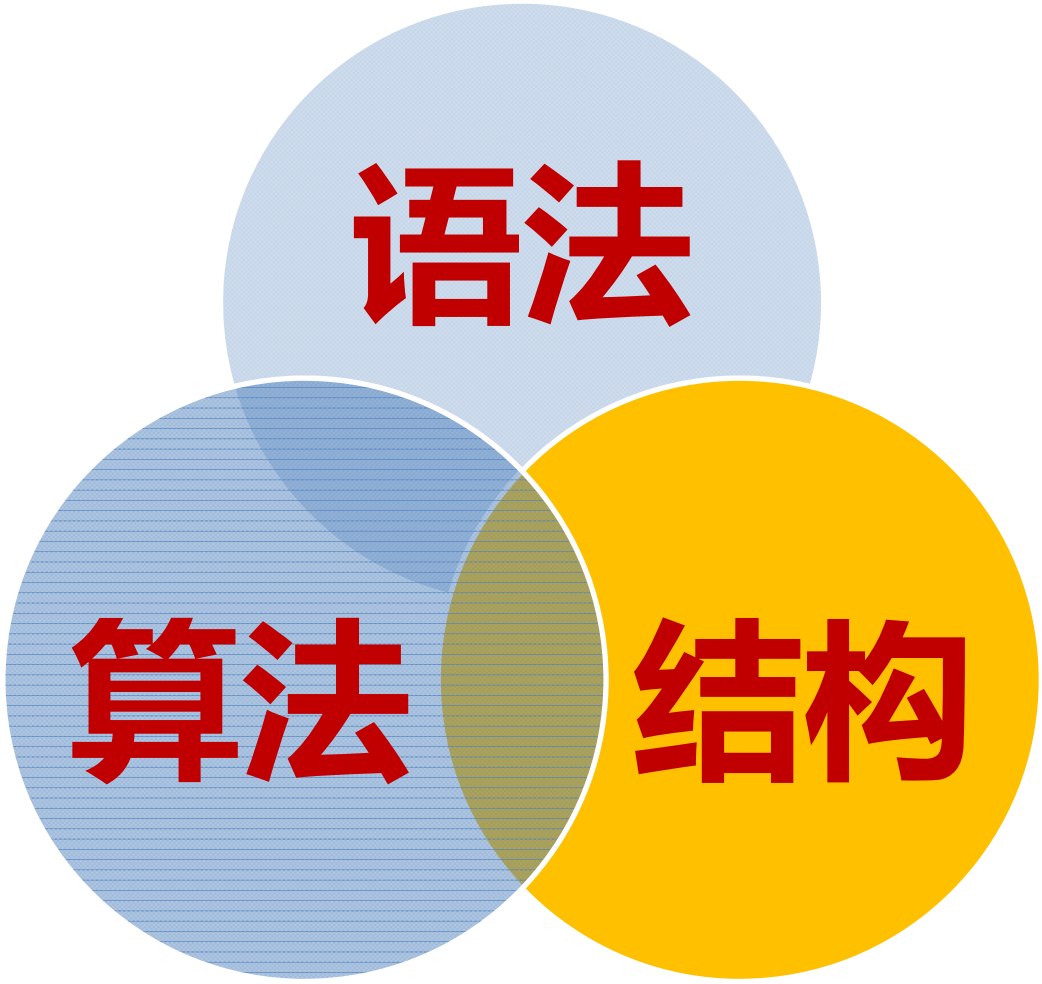
```
char *strcpy(char *s1, const char *s2);
char *strcat(char *s1, const char *s2);
int strcmp(const char *s1, const char *s2);
size_t strlen(const char *cs);
```

```
srand(seed);
rand( );
```

常用数学函数：

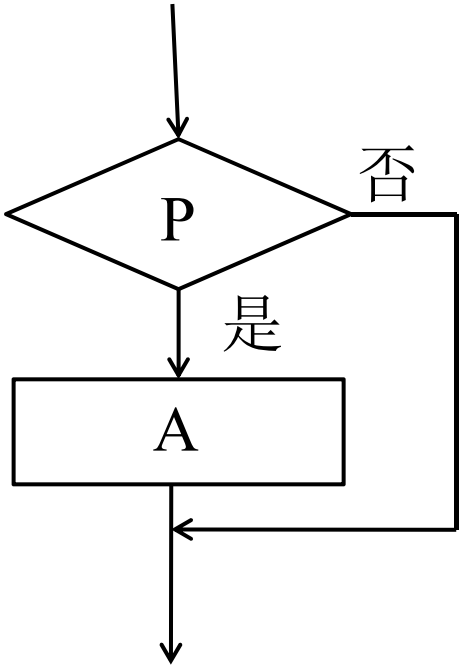
fabs(double); sqrt(double);……

总结

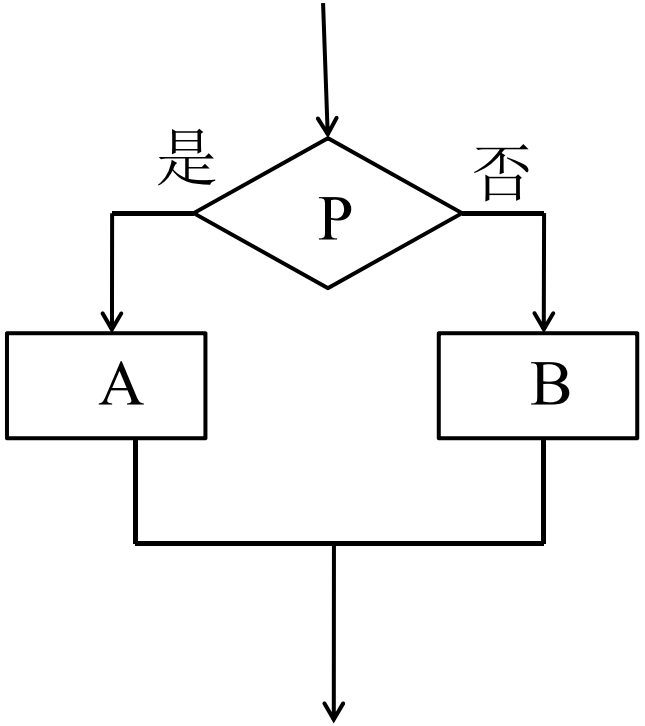


逻辑控制结构

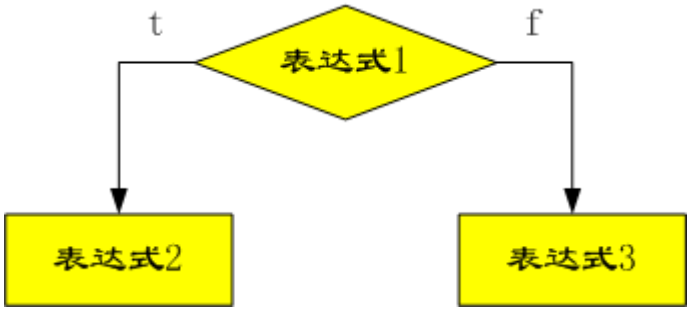
- 顺序
- 分支
- 循环
- 递归

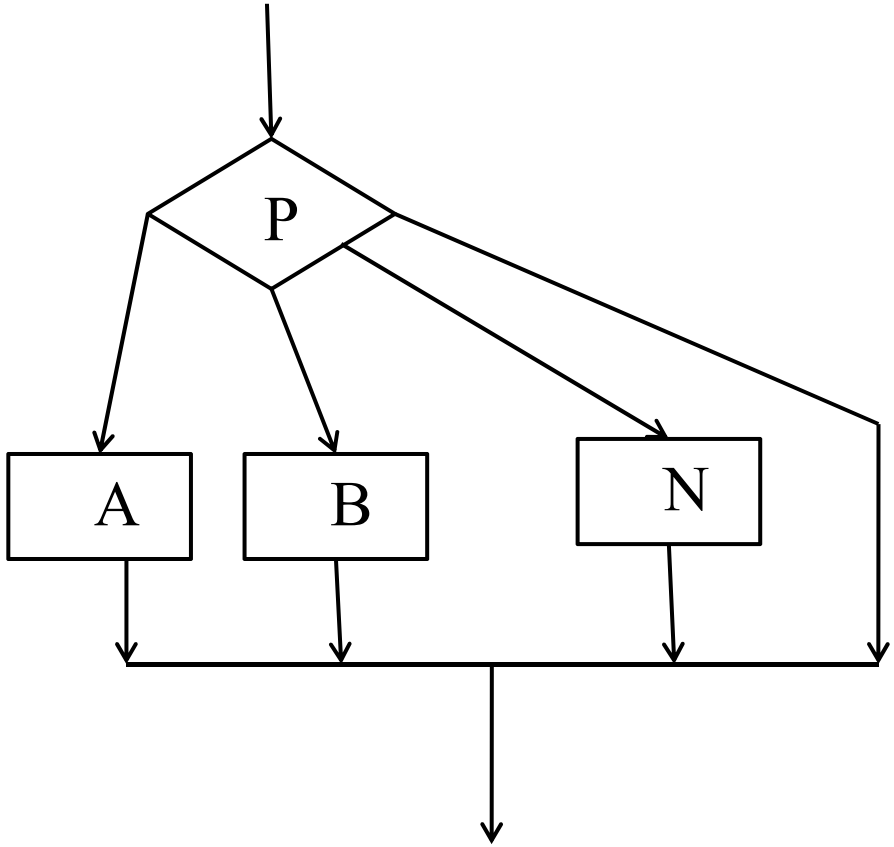


单分支

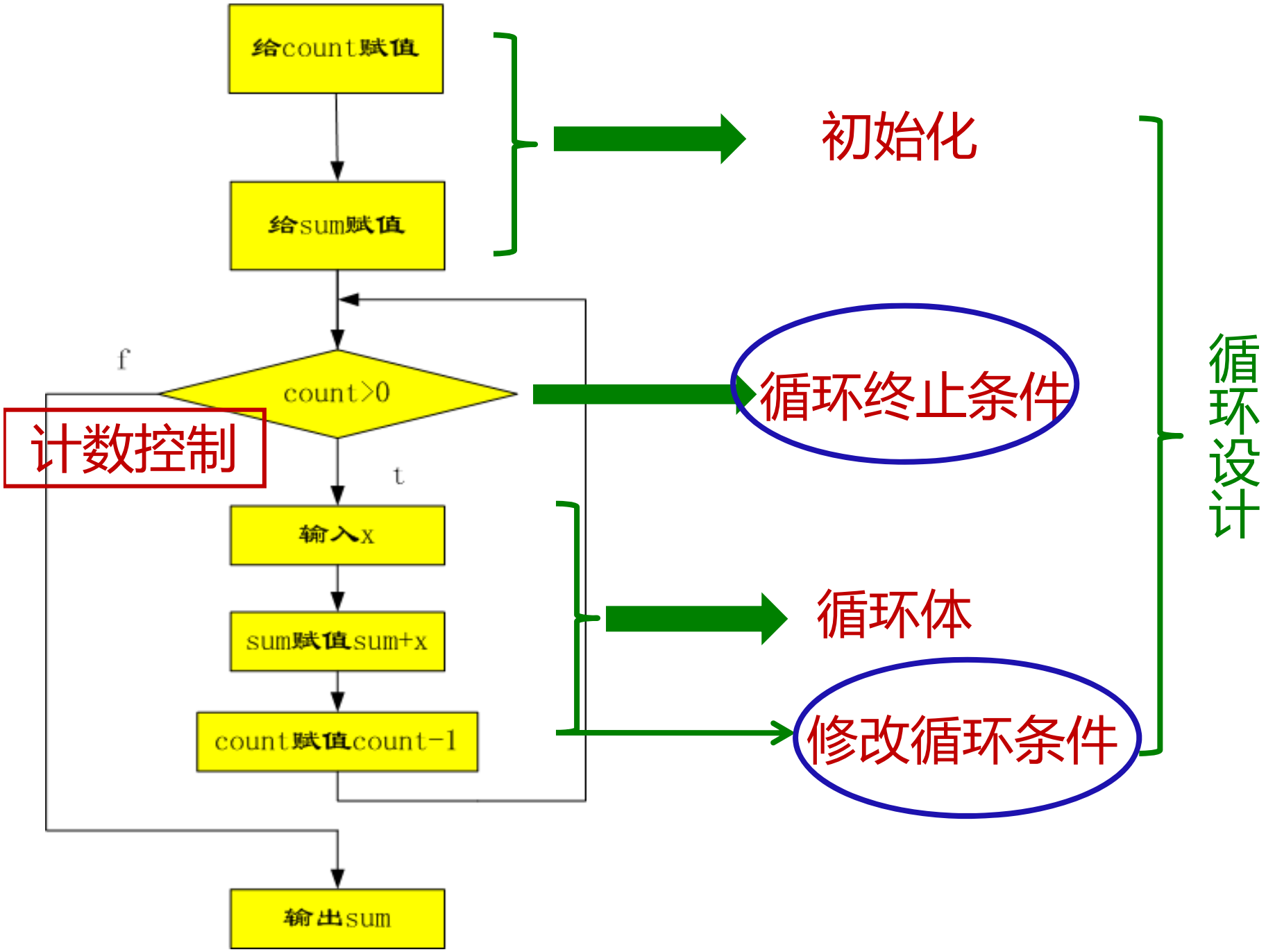


双分支

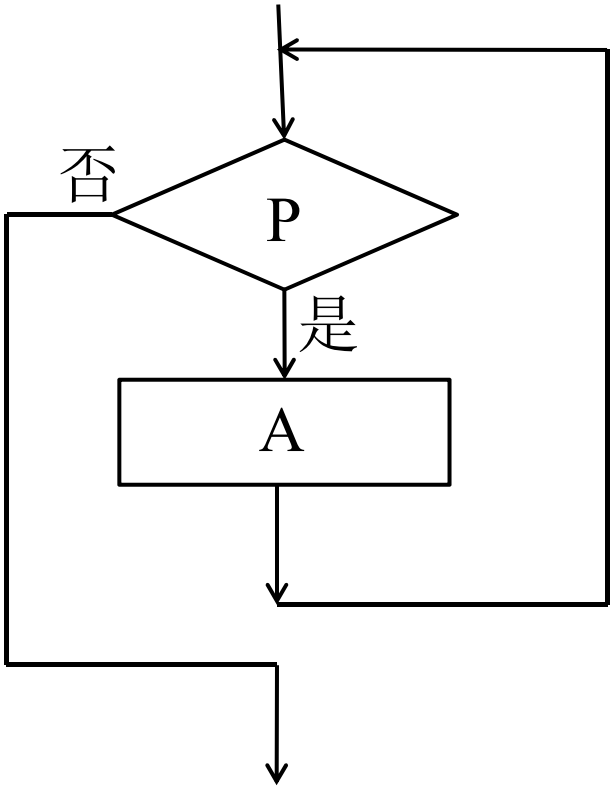




多分支


给count赋值
给sum赋值
初始化
f
count>0
循环终止条件
计数控制
t
输入x
sum赋值sum+x
循环体
count赋值count-1
修改循环条件
循环设计
输出sum

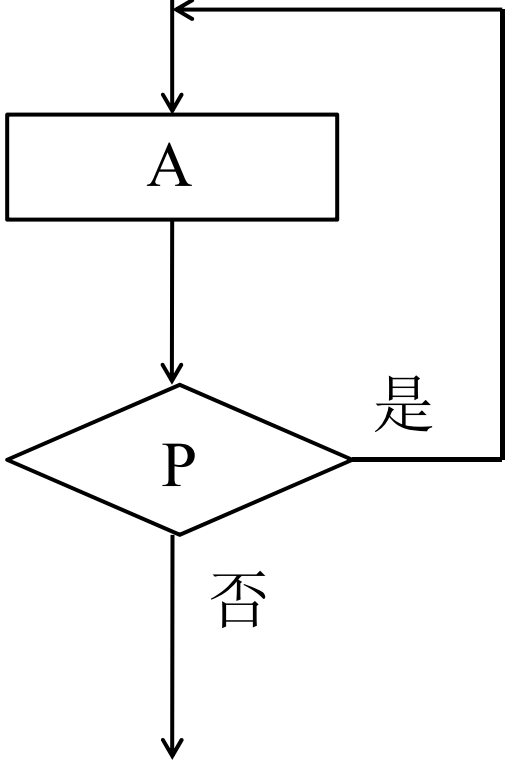
while (表达式)
循环体语句



当型循环

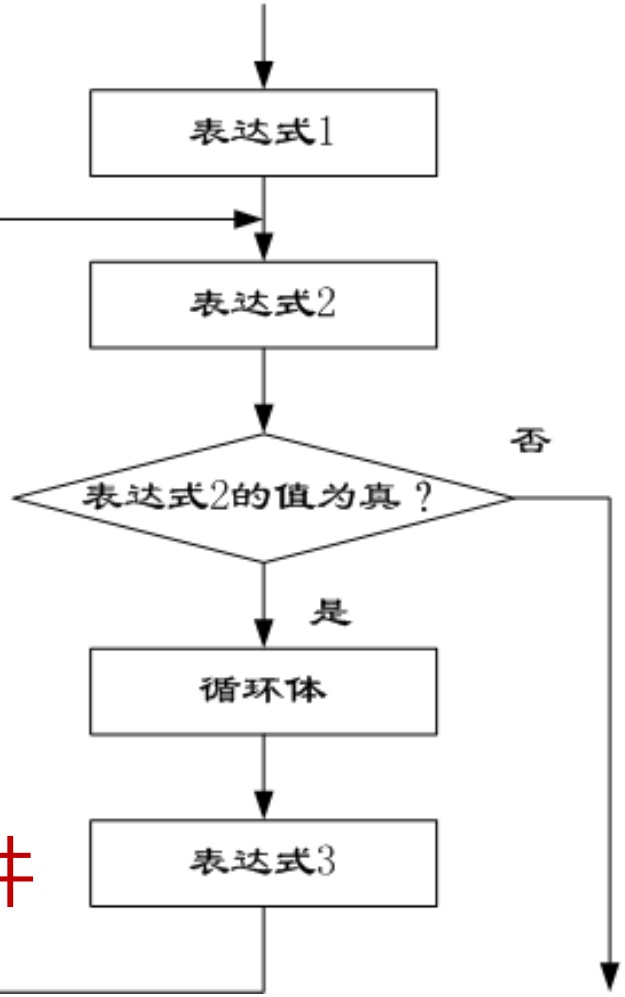
do 循环体语句
while(表达式)；

至少执行一次循环体



直到型循环

初始化



终止条件

修改循环条件

递归算法的一般形式：

```
返回类型   函数名( 参数)
{
    if(结束条件){
        基本项 ;      // 递归出口
    }
    else
        {
            归纳项 ;
            调用本身，递归 ;
        }
    return 表达式;
}
```

```
int fac(int n)
{
        int y;
        if (n==0||n==1)
                y=1;
        else
                y=n*fac(n-1);
        return  y;
}
```

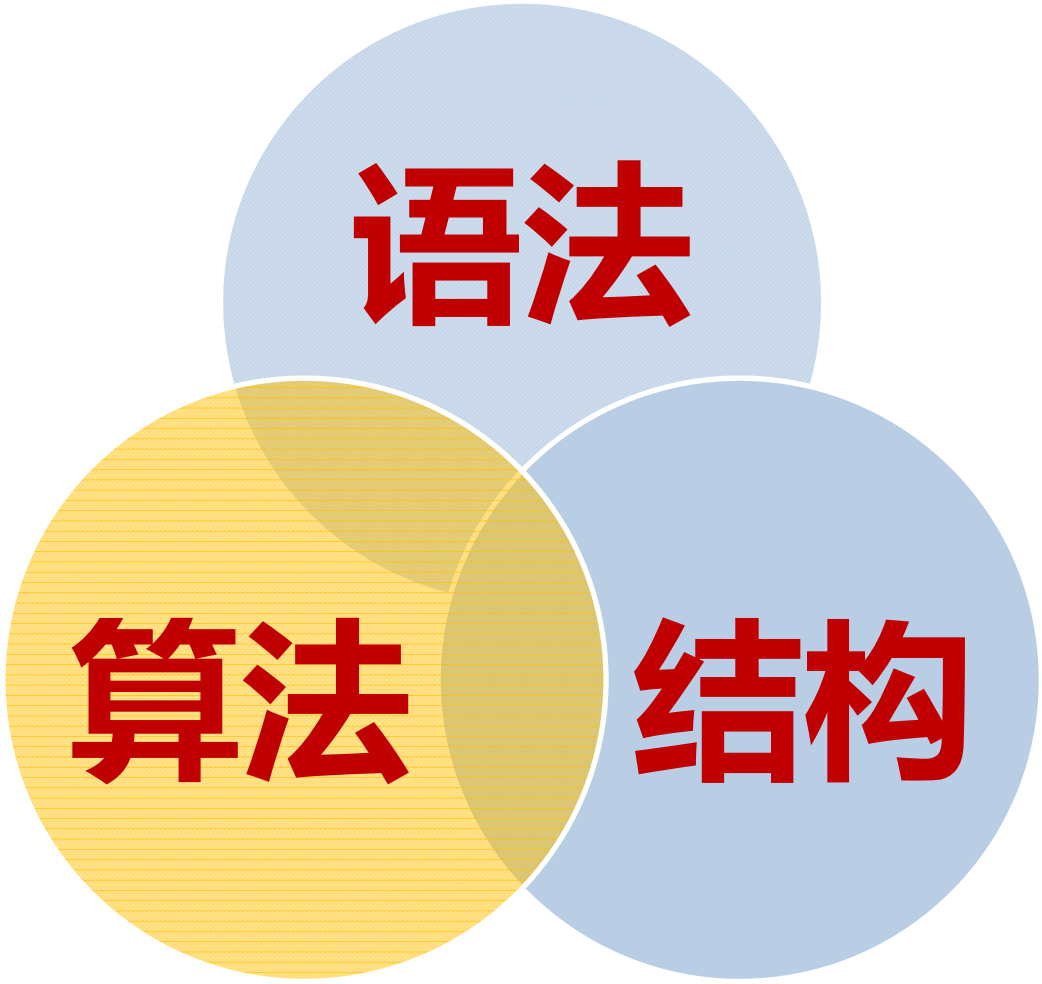
递归与迭代

递归方法	迭代方法
f(n)=n*f(n-1) f(0)=1	f(n)=n*(n-1)*…*1 f(0)=1

```
int fac(int n)
{
        int y;
        if (n==0||n==1)
                y=1;
        else
                y=n*fac(n-1);
        return   y;
}
```

```
int fac_it(int n)
{
   int count, result=1;
   for(count=1; count<=n; count++)
        result*=count;
   return result;
}
```

总结



算法

- 穷举法
- 递推法
- 迭代法
- 其他(辗转相除法，...)
- 查找

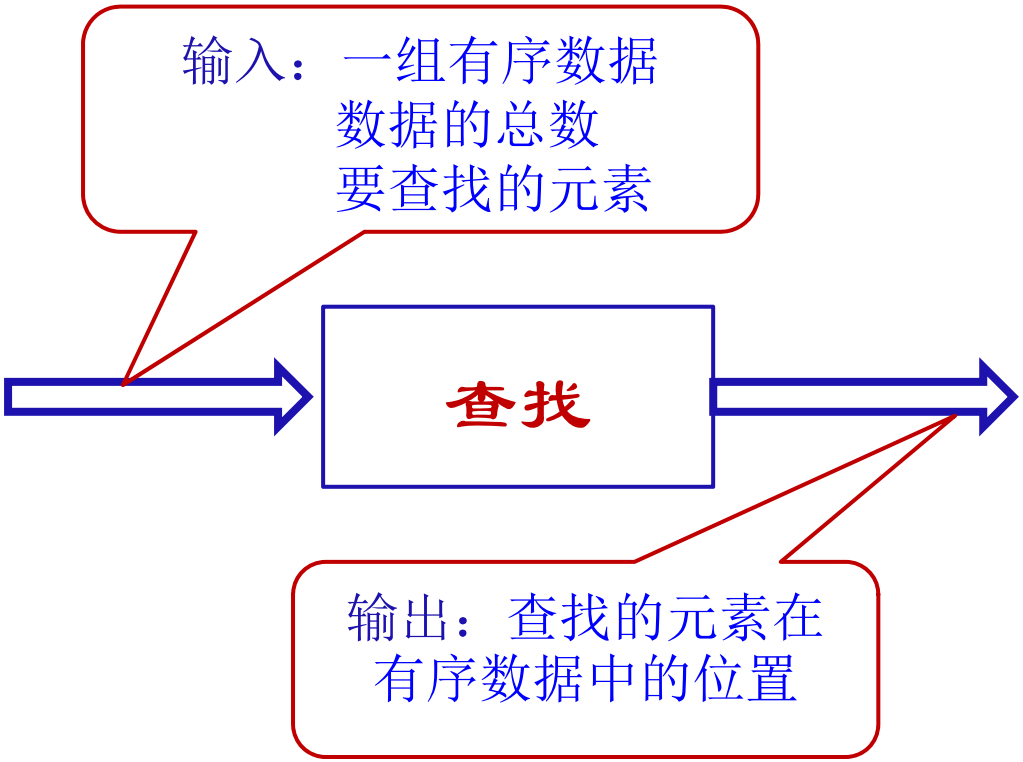
- 常用算法
  - 直接法
    - 求素数，多进制数之间的转换，数的逆序输出。。。
  - 递推法(迭代法)
    - 裴波那契数列，迭代法求近似值。。。
  - 穷举法
    - 水仙花数，百鸡问题，候选人。。。

- 文件（文本文件）操作---四部曲
  - 读文件
    - 建立；打开；读；关闭；
  - 写文件
    - 建立；打开；写；关闭；

查找

顺序查找

二分查找




```
int l=0,r=N-1,m;
while(l<=r)
{      m =(l+r)/2;
       if(a[m]>x)   r= m-1;
                  else if(a[m]<x) l= m+1;
                    else {cout<<m; break；  }
       }
```

预祝大家考试顺利!