C++程序编写——初步

程龚 gcheng@nju.edu.cn

- 程序(program)是什么？

计算一些水果的总价

算法

1. 初始时，总计为0。
2. 对每一种水果，小计它的总价，并累加到总计中。
3. 最终结果就是总计。

算法

1. 初始时，总计为0。
2. 对每一种水果，小计它的总价，并累加到总计中。具体而言：
   ① 获得这种水果的单价。
   ② 获得这种水果的数量。
   ③ 小计=单价x数量。
   ④ 新的总计=旧的总计+小计。
3. 最终结果就是总计。

~~自然语言~~

程序语言

将算法编写为程序

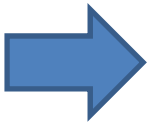
1. Choose a Compression Ratio.
2. Choose a Threshold Value.
3. Calculate the Sentence Number Quota of the summary.
4. Divide the document into range blocks.
5. Transform the document into fractal tree.
6. Set the current node to the root of the fractal tree.
7. **Repeat**

7.1 **For** each child node under current node,
    Calculate the fractal value of child node.

7.2 Allocate Quota to child nodes in proportion to fractal values.

7.3 **For** each child nodes,
  If the quota is less than threshold value
    Select the sentences in the range block by summarization
  Else
    Set the current node to the child node
    Repeat Step 7.1, 7.2, 7.3

8. **Until** all the child nodes under current node are processed



```
/* Park-Miller "minimal standard" 31 bit
 * pseudo-random number generator, implemented
 * with David G. Carta's optimisation: with
 * 32 bit math and without division.
 */

long unsigned int rand31_next()
{
    long unsigned int hi, lo;

    lo = 16807 * (seed & 0xFFFF);
    hi = 16807 * (seed >> 16);

    lo += (hi & 0x7FFF) << 16;
    lo += hi >> 15;

    if (lo > 0x7FFFFFFF) lo -= 0x7FFFFFFF;

    return ( seed = (long)lo );
}
```

自然语言

程序语言

- 程序设计=算法设计+程序编写

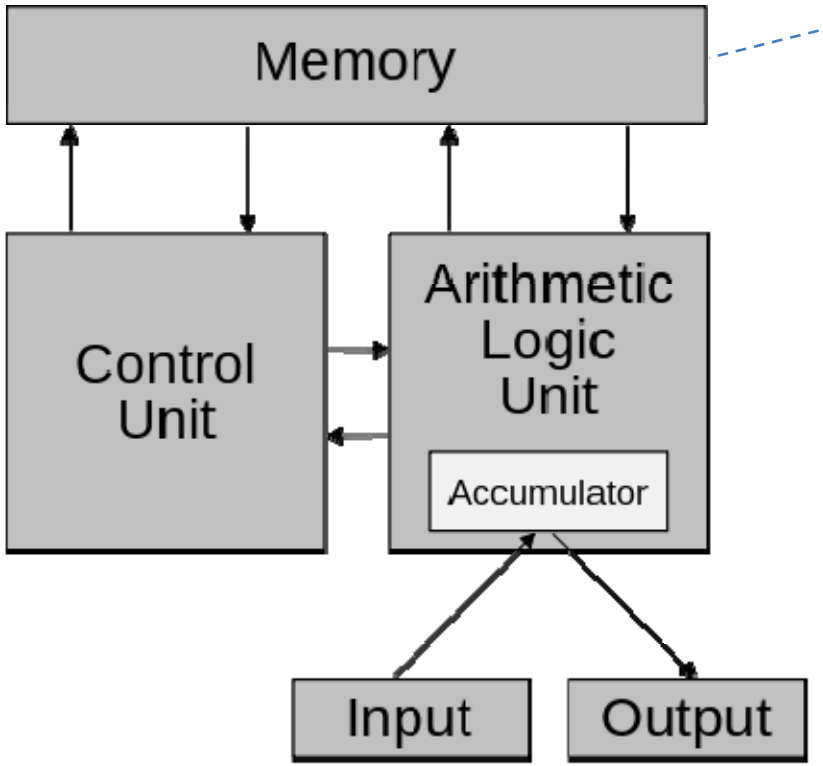
- 编程=翻译
- C++只是程序语言中的一种

程序：计算机可以执行的一系列指令

```
/* Park-Miller "minimal standard" 31 bit
 * pseudo-random number generator, implemented
 * with David G. Carta's optimisation: with
 * 32 bit math and without division.
 */

long unsigned int rand31_next()
{
    long unsigned int hi, lo;

    lo = 16807 * (seed & 0xFFFF);
    hi = 16807 * (seed >> 16);

    lo += (hi & 0x7FFF) << 16;
    lo += hi >> 15;

    if (lo > 0x7FFFFFFF) lo -= 0x7FFFFFFF;

    return ( seed = (long)lo );
}
```

- 计算机可以做什么？

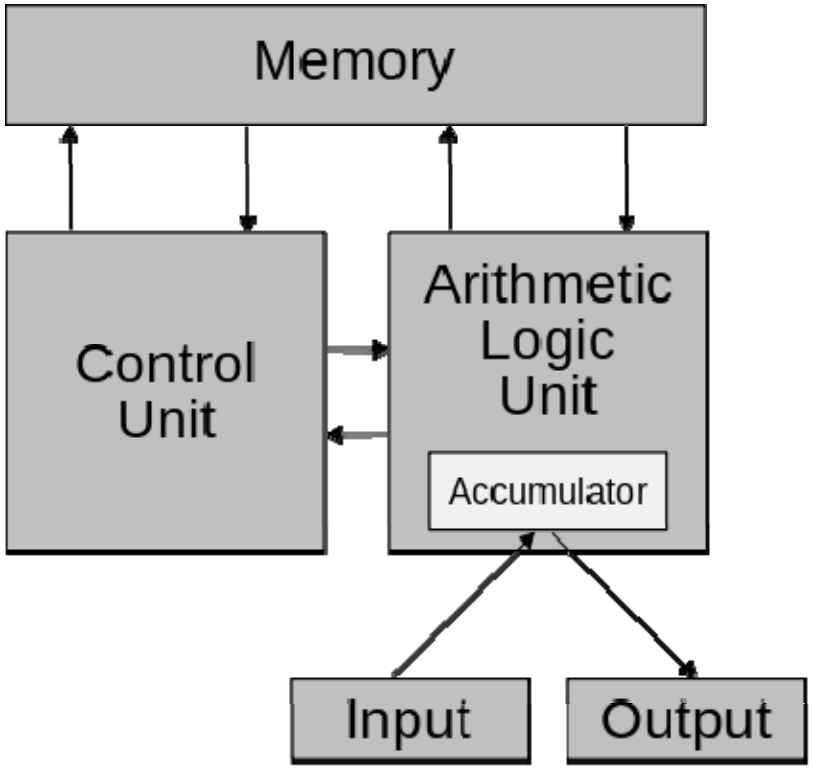
von Neumann体系结构



地址	内容
…	…
2430	10110000
2431	11110000
2432	00000000
2433	00010100
2434	01001111
…	…

(von Neumann)计算机可以做什么

- 存储
- 运算
- 输入
- 输出
- ……



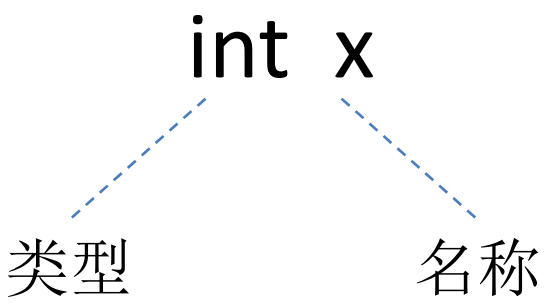
1、变量 (variable)

- 变量：存储单元的名称
- 需要使用存储单元时，就声明一个变量
- 变量必须先声明后使用

	地址	内容
	…	…
x	2430	10110000
	2431	11110000
	2432	00000000
y	2433	00010100
	2434	01001111
	…	…

1、变量 (cont.)

- 变量的声明(declaration)

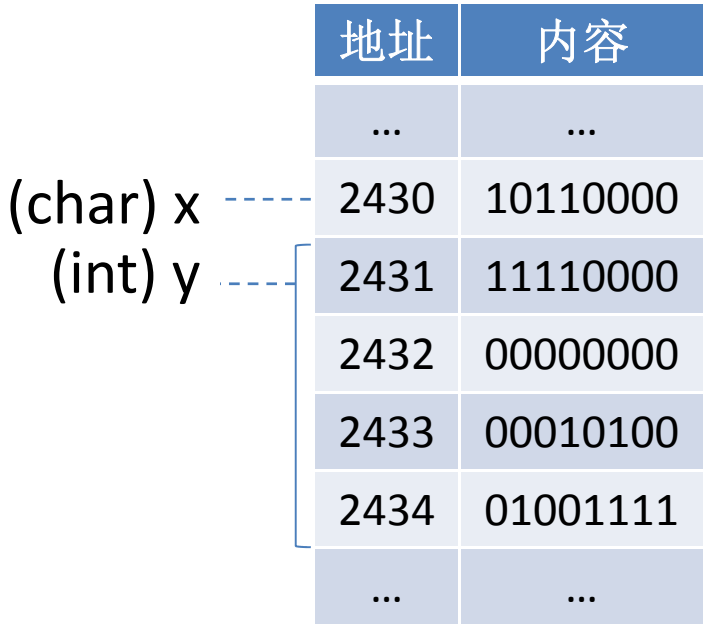


1、变量 (cont.)

- 变量的名称
  - 首字符：字母、下划线
  - 后续字符：字母、数字、下划线
- 举例
  - 合法的变量名称：x x1 x_1 _abc ABC123z7 sum RATE count data2 Big_Bonus
  - 非法的变量名称：12 3X %change data-1 myfirst.c PROG.CPP
- 注意
  - 大小写敏感：abc ABC
  - 保留词：int if
  - 有意义的名称：x y → width height

1、变量 (cont.)

- 变量的类型，决定了
  - 占用的存储单元的数量
  - 内容的编码方法

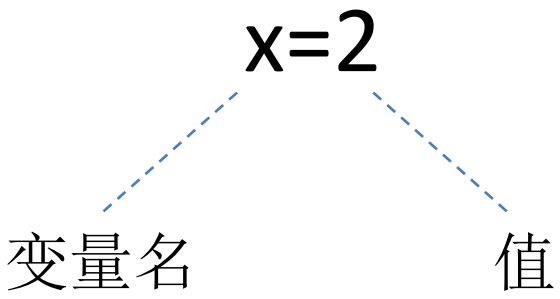


1、变量 (cont.)

- 变量类型的例子
  - 布尔：bool
  - 数值：int long double
  - 字符：char

1、变量 (cont.)

- 变量的赋值(assignment)



(char) x
(int) y

地址	内容
…	…
2430	10110000
2431	11110000
2432	00000000
2433	00010100
2434	01001111
…	…

1、变量 (cont.)

- 变量的声明时赋值

```
int  x=2
```

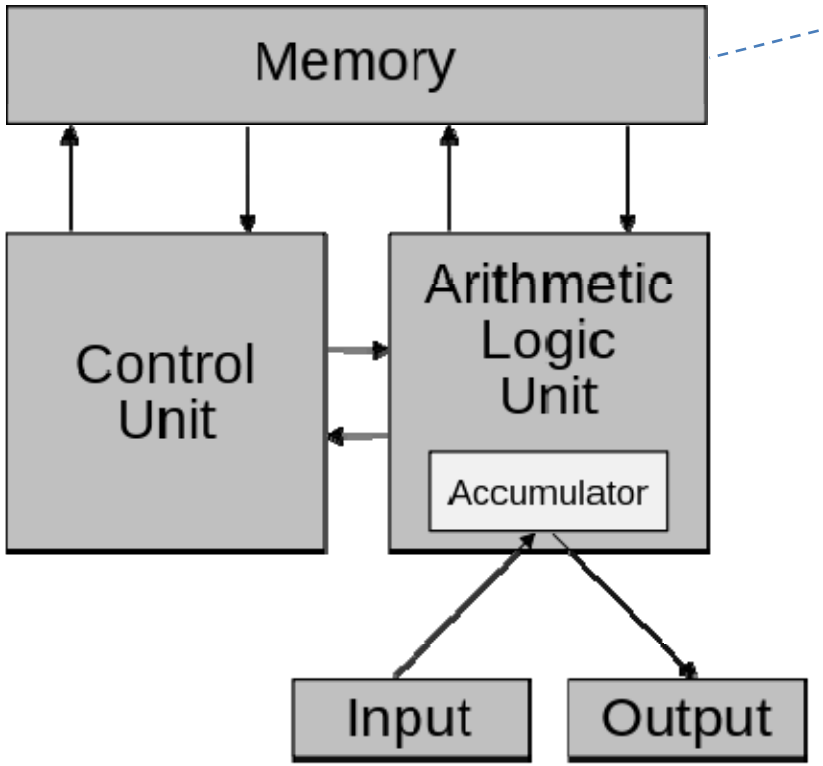
2、常量 (constant)

- 常量的类型是不言自明的
  - 布尔：true false
  - 数值：123 45.67 89L
  - 字符：’a’ ‘\n’
- （不严格而言，）常量不占用存储单元
  - 因此，不能给常量赋值

编程举例

1. 初始时，总计为0。(int total=0)
2. 对每一种水果，小计它的总价，并累加到总计中。具体而言：
   ① 获得这种水果的单价。(int price=2)
   ② 获得这种水果的数量。 (int quantity=3)
   ③ 小计=单价x数量。(int subTotal)
   ④ 新的总计=旧的总计+小计。
3. 最终结果就是总计。

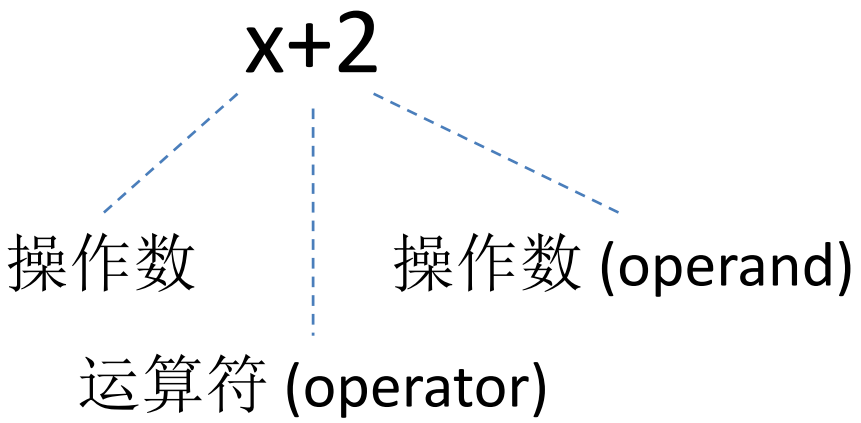
3、运算/表达式 (expression)



地址	内容
…	…
2430	10110000
2431	11110000
2432	00000000
2433	00010100
2434	01001111
…	…

3、表达式 (cont.)

- 表达式的构成



3、表达式 (cont.)

- 运算符作用的操作数的数量
  - 一元(unary)表达式：x++ !y
  - 二元(binary)表达式：x+y x&&y

3、表达式 (cont.)

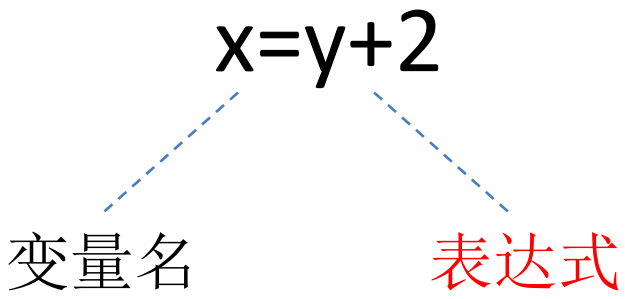
- 运算符的例子
  - 数值：+ - * / % ++
  - 布尔：！&& > <=
  - 位：~ << |
  - 赋值：=

3、表达式 (cont.)

- 表达式的值(value)：运算的结果
  - 当x是3时，x+2的值是5
- 表达式的值具有类型
  - 当x和y都是int时：x+y的值也是int
- 当两个操作数的类型不一致时，会自动进行类型的转换
  - 当x是double、y是int时：y先被自动转换成double，从而x+y的值也是double

3、表达式 (cont.)

- 变量的赋值



- 变量和常量是无运算符的特殊表达式

3、表达式 (cont.)

- 复合(compound)表达式：x+y*2
- 运算符有优先级(precedence)
- 改变运算符的优先级：(x+y)*2

编程举例

1. 初始时，总计为0。(int total=0)
2. 对每一种水果，小计它的总价，并累加到总计中。具体而言：
   ① 获得这种水果的单价。(int price=2)
   ② 获得这种水果的数量。 (int quantity=3)
   ③ 小计=单价x数量。(int subTotal=price*quantity)
   ④ 新的总计=旧的总计+小计。(total=total+subTotal)
3. 最终结果就是总计。

4、语句 (statement)

- 程序：计算机可以执行的一系列指令
- 语句是程序的主要组成部分

4、语句 (cont.)

- 声明语句

```
int  x;
```

4、语句 (cont.)

- 赋值语句

```
x=y+2;
```

4、语句 (cont.)

- 表达式语句

```
y+2;
```

4、语句 (cont.)

- 空(null)语句

;

- 用途：语法上需要一个语句，但无事可做

4、语句 (cont.)

- 复合(compound)语句

```
{
  int x=2;
  x=4+2;
}
```

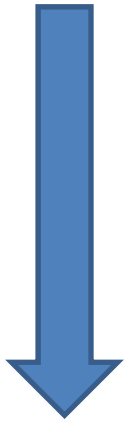
- 用途：语法上需要一个语句，但事情太多

编程举例

1. 初始时，总计为0。(int total=0;)
2. 对每一种水果，小计它的总价，并累加到总计中。具体而言：
   ① 获得这种水果的单价。(int price=2;)
   ② 获得这种水果的数量。 (int quantity=3;)
   ③ 小计=单价x数量。(int subTotal=price*quantity;)
   ④ 新的总计=旧的总计+小计。(total=total+subTotal;)
3. 最终结果就是总计。

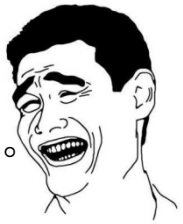
程序：计算机可以执行的一系列指令

- 通常情况下，程序只能逐条语句顺序执行



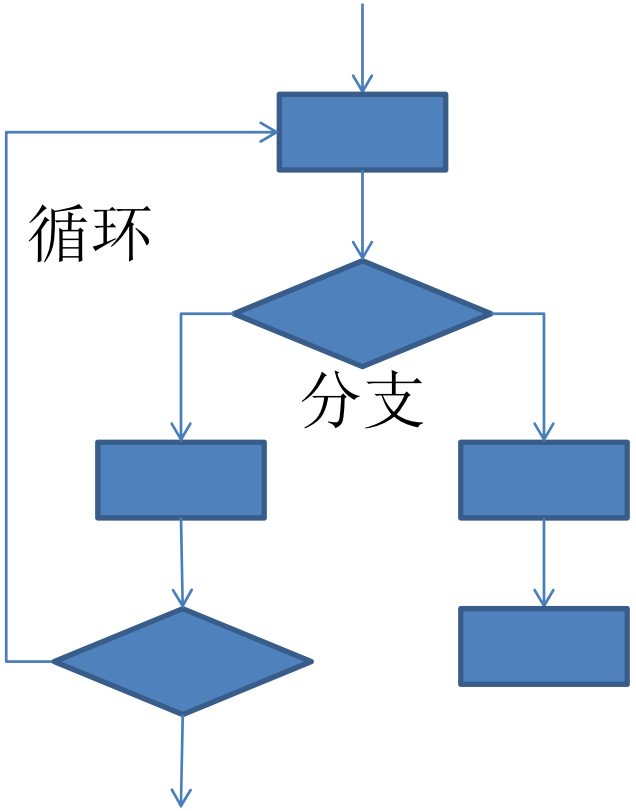
```
int total=0;
int price=2;
int quantity=3;
int subTotal=price*quantity;
total=total+subTotal;
```

对每一种水果，小计它的总价，并累加到总计中。



5、流程控制 (flow of control)

- 分支：有条件地执行一个语句
- 循环：重复地执行一个语句



5、流程控制 (cont.)

- if语句

if (*条件*)
  执行这一条语句

- *条件*是一个值为布尔类型的表达式
  – 例如：x+y>2
- *条件*的值是true时，才执行一条特定的语句

5、流程控制 (cont.)

- if-else语句

```
if (条件)
  执行这一条语句
else
  执行这一条语句
```

- *条件*的值是true或false时，执行不同的语句

5、流程控制 (cont.)

- 复合语句的作用

if (*条件*)
  执行这一条复合语句
else
  执行这一条复合语句

5、流程控制 (cont.)

- 复合语句的作用

```
if (条件) {
  执行这些语句
} else {
  执行这些语句
}
```

5、流程控制 (cont.)

- 嵌套的if-else语句

if (*条件*)
  执行这一条语句
else
  执行这一条if-else 语句

5、流程控制 (cont.)

- 嵌套的if-else语句

if (*条件1*)
  执行这一条语句
else if (条件2)
  执行这一条语句
else
  执行这一条语句

5、流程控制 (cont.)

- if-else语句的例子

```
if (x==1)
  y=5;
else if (x==2) {
  y=x+z;
  z=5;
} else
  z=x;
```

5、流程控制 (cont.)

- switch语句：代替深度嵌套的if-else语句

```
switch (表达式) {
  case 值1: 执行这些语句
  case 值2: 执行这些语句
  case 值3: 执行这些语句
  ……
}
```

- *表达式*的值与某个case的值相同时，从该case开始顺序执行后续所有语句

5、流程控制 (cont.)

- switch语句的例子

```
switch (x) {
  case 1: y=5;
  case 2: y=x+z;
          z=5;
  case 3: z=x;
}
```

5、流程控制 (cont.)

- break语句：中止switch语句的执行

```
switch (表达式) {
  case 值1: 执行这些语句
            break;
  case 值2: 执行这些语句
            break;
  case 值3: 执行这些语句
            break;
  ……
}
```

5、流程控制 (cont.)

- break语句的例子

```
switch (x) {
  case 1: y=5;
          break;
  case 2: y=x+z;
          z=5;
          break;
  case 3: z=x;
          break;
}
```

5、流程控制 (cont.)

- default：缺省情况

```
switch (表达式) {
  case 值1: 执行这些语句
            break;
  case 值2: 执行这些语句
            break;
  case 值3: 执行这些语句
            break;
  default: 没有任何case匹配时，执行这些语句
}
```

5、流程控制 (cont.)

- default语句的例子

```
switch (x) {
  case 1: y=5;
          break;
  case 2: y=x+z;
          z=5;
          break;
  default: z=x;
}
```

```
if (x==1)
  y=5;
else if (x==2) {
  y=x+z;
  z=5;
} else
  z=x;
```

5、流程控制 (cont.)

- while语句

while (*条件*)
  执行这一条语句

- 只要*条件*的值是true，就重复执行一条特定的语句

5、流程控制 (cont.)

- while语句的例子

```
while (x>0)
  x=x-1;
```

5、流程控制 (cont.)

- 避免死循环

```
while (x>0)
  y=y-1;
```

5、流程控制 (cont.)

- 其它可以实现循环的语句（自学）
  - for语句
  - do-while语句

5、流程控制 (cont.)

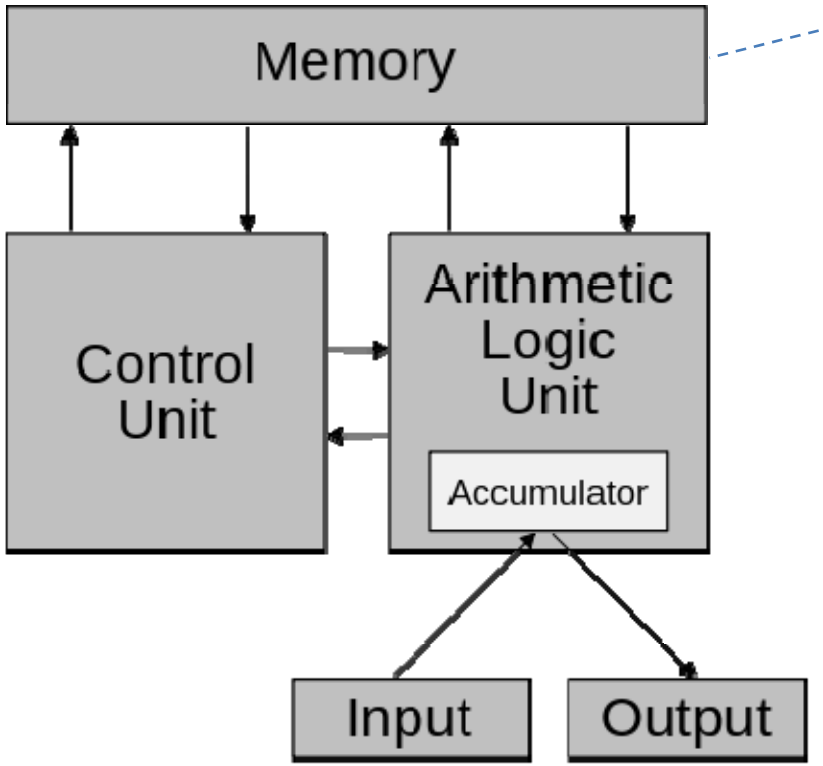
- continue语句：跳过循环体内的后续语句

```
while (x>0) {
  x=x-1;
  if (x!=2)
    continue;
  y=y+1;
}
```

编程举例

```
int total=0;
while (还有水果未计算) {
  int price=2;
  int quantity=3;
  int subTotal=price*quantity;
  total=total+subTotal;
  将这种水果标记为“已计算”
}
```

6、输入/输出 (input/output)



地址	内容
…	…
2430	10110000
2431	11110000
2432	00000000
2433	00010100
2434	01001111
…	…

6、输入/输出 (cont.)

- cin语句：用键盘的输入给变量赋值

```
int x;
cin >> x;
int y;
double z;
cin >> y >> z;
```

6、输入/输出 (cont.)

- cout语句：将表达式的值输出到屏幕

```
int x=2;
cout << “x的值是\n” << x;
```

```
x的值是
2
```

编程举例

```
int total=0;
while (还有水果未计算) {
  int price=2;
  int quantity=3;
  int subTotal=price*quantity;
  total=total+subTotal;
  将这种水果标记为“已计算”
}
cout << total;
```

- 对于一个复杂的问题，如何设计算法、编写程序？
  - 策略1：逐步设计、编写
  - 策略2：自顶向下设计、编写

算法——顶层设计

1. 初始时，总计为0。
2. 对每一种水果，小计它的总价，并累加到总计中。
3. 最终结果就是总计。

算法——下层设计

1. 初始时，总计为0。
2. 对每一种水果，小计它的总价，并累加到总计中。具体而言：
   ① 获得这种水果的单价。
   ② 获得这种水果的数量。
   ③ 小计=单价x数量。
   ④ 新的总计=旧的总计+小计。
3. 最终结果就是总计。

7、函数 (function)

- 将总体的plan逐步分解细化为若干小的plan
- 将总体的程序逐步划分为若干子程序，称作函数

编程举例

```
int total=0;
while (还有水果未计算) {
  int subTotal = computeSubTotal(这种水果);
  total=total+subTotal;
  将这种水果标记为“已计算”
}
```

调用函数（读取运算结果）

运算结果的类型　　名称　　参数

```
int computeSubTotal (一种水果) {
  int price=2;
  int quantity=3;
  int subTotal=price*quantity;
  return subTotal;
}
```

返回运算结果

集成开发环境 (IDE)

- 更方便、更有效地开发程序
  - 管理（例如：按项目组织程序文件）
  - 编写（例如：代码输入提示）
  - 调试（例如：监控变量值）

用Visual Studio编写第一个C++程序

1. 菜单→文件→新建→项目→Win32控制台应用程序→为项目命名→设置为空项目
2. 解决方案资源管理器→源文件→(右键)添加→新建项→C++文件→为文件命名
3. 录入DISPLAY 1.10
4. (F7)生成解决方案→观察输出窗口是否生成成功
5. 如果生成成功（程序没有语法错误）
   - 设置/取消断点：F9
   - 调试（在断点处暂停）：F5
   - 调试（一步一停）：F10/F11
   - 执行（不停）：Ctrl+F5