

一、结构化程序设计

【大纲内容】

【1】顺序结构、分支结构和循环结构

【2】自顶向下、逐步求精的模块化程序设计

【2】流程图的概念及流程图描述

(一) 程序三大基本结构 (Program Control Structures)

在编程中，我们经常会安排程序按照一定的“路线”执行任务，这些路线就叫程序结构 (Program Structure) 。

C++ 中有三种最基本的结构：

1、顺序结构 (Sequential Structure)

按照从上往下、一条一条语句顺序执行的方式。

像你早上起床后，一步一步完成：洗脸 → 刷牙 → 吃早饭，不能跳过。

C++ 示例：

```
cout << "打开书包" << endl;
cout << "拿出作业本" << endl;
cout << "开始写作业" << endl;
```

2、分支结构 (Selection / Conditional Structure)

程序会根据某个条件，选择不同的路径。

如果今天下雨就打伞，不下雨就骑车。程序也会根据条件来判断。

C++ 示例：

```
if (isRaining) {
    cout << "带伞出门" << endl;
} else {
```

```
cout << "骑车出门" << endl;

}
```

3、循环结构 (Loop Structure)

当需要重复做某件事情时，用循环结构。

像做 10 道练习题，每道题都要写，就用循环。

C++ 示例：

```
for (int i = 1; i <= 5; i++) {

    cout << "第" << i << "遍背课文" << endl;

}
```

考试注意事项 & 易错点：

错误类型	正确做法
结构写反了	顺序写在前，分支要判断，循环重复
忘了加大括号	分支或循环中多个语句要加 {}
判断条件写错	== 是等于判断，别用成 =

(二) 模块化程序设计：自顶向下，逐步求精 (Top-down, Stepwise Refinement)

模块化设计 (Modular Design) 是将**程序分成一个个功能小模块**，每个模块只管一件事。

- “自顶向下” 就是：先想总体框架 → 再分小任务 → 每个任务继续分小步骤。
- “逐步求精” 就是：先写简单的外壳 → 一点点补充详细内容。

就像你写作文：

- 先写题目和大纲 (自顶向下)
- 再一段一段慢慢补充内容 (逐步求精)

或者像拼乐高：

- 先搭底座 → 再搭墙 → 再搭窗户 → 最后装饰

C++ 示例：

```

void openBook() {
    cout << "打开书本" << endl;
}

void doHomework() {
    cout << "开始写作业" << endl;
}

int main() {
    openBook();    // 主程序先调用小模块
    doHomework(); // 每个模块负责一件事
    return 0;
}

```
















考试注意事项 & 易错点：

错误点	正确方法
把所有代码都写在一起	要把功能拆成多个小模块（函数）
模块顺序乱了	先总后细，先写框架再写内容
函数没加 () 或 return	每个函数都要完整定义与调用

(三) 流程图的概念及描述 (Flowchart)

流程图 (Flowchart) 是用图形符号画出的程序执行路线图，用来清楚地表示程序的步骤和判断。

常见流程图符号：

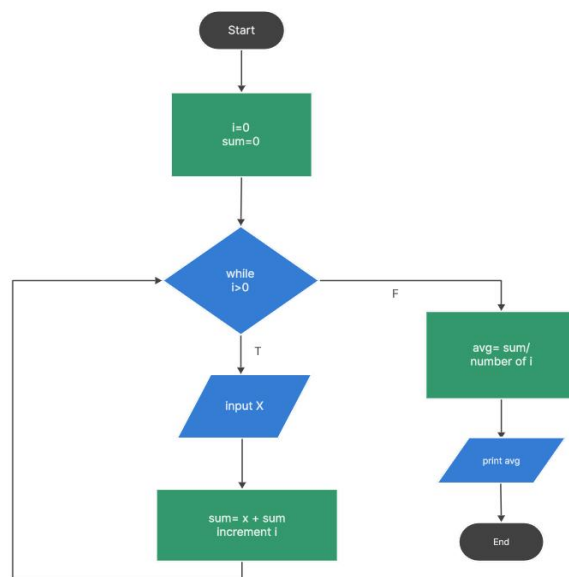
名 称	符 号	名 称	符 号	名 称	符 号
起止框		多重文档		单向连接线	
处理框		子流程		双向连接线	
判断框		文档		手动输入	
输出/输入框		卡片		数据存储	
或者		总和		角色	 Actor

示例流程图：判断奇偶数

文字描述：

- 开始：初始化两个变量 $i = 0$ 和 $sum = 0$ 。
- 输入与计算：
 - 进入一个 while 循环，条件是 $i > 0$ 。如果不满足条件，退出循环并计算平均值。
 - 如果 $i > 0$ ，则执行以下操作：
 - ◆ 输入一个数字 X 。
 - ◆ 将 X 加到 sum 上，并使 i 增加 1。
- 结束：
 - 计算并打印出平均值 $avg = sum / i$ 。

流程图



C++ 对应代码：

```

#include <iostream>

using namespace std;

int main() {
    int i = 0, sum = 0, x;

    while (i > 0) {
        cin >> x;

        sum += x;

        i++;
    }

    // 计算并打印平均值
    double avg = (i == 0) ? 0 : static_cast<double>(sum) / i;

    cout << "Average: " << avg << endl;

    return 0;
}
  
```

考试易错点：**错误点****正确做法**

流程图不封闭

要有“开始”和“结束”节点

判断不清楚

菱形出口要标明“是 / 否”

箭头混乱

每一步都必须**有清晰箭头指向**