焦作市第二十六中学

信息学奥赛校本教材

（上册）

焦作市第二十六中学信奥社团

目录

Noip Blocky 简介 1
第一章 语法基础 6
1.1 C++输入与输出 7
1.2 变量与算术运算 12
1.3 条件与分支 17
1.4 while 循环 24
1.5 For 循环 28
1.6 多重循环 31
1.7 循环应用（1） 37
1.8 循环应用（2） 41
1.9 循环应用（3） 45

Noip Blocky 简介

1. 什么是 Noip？

全国青少年信息学奥林匹克联赛（NOIP）是一项面向全国青少年的信息学竞赛和普及活动，旨在向那些在中学阶段学习的青少年普及计算机科学知识；给学校的信息技术教育课程提供动力和新的思路；给那些有才华的学生提供相互交流和学习的机会；通过竞赛和相关的活动培养和选拔优秀的计算机人才。
2010 年 11 月 19 日，教育部宣布取消了各项奥林匹克竞赛全国决赛一等奖以下的高校保送资格，改由所在地招生委员会决定是否给予 20 分及以下的加分。
2020 年强基计划发布，高考通过竞赛保送已被严格控制。只有在 5 大学科竞赛中拿到前 50 名进入国家集训队的学生才可以直接保送清北等名校。其他奖项则需通过高校组织的校考获得低于高考分数线入取的资格。

2. 为什么要用 Noip Blocky 编辑器？

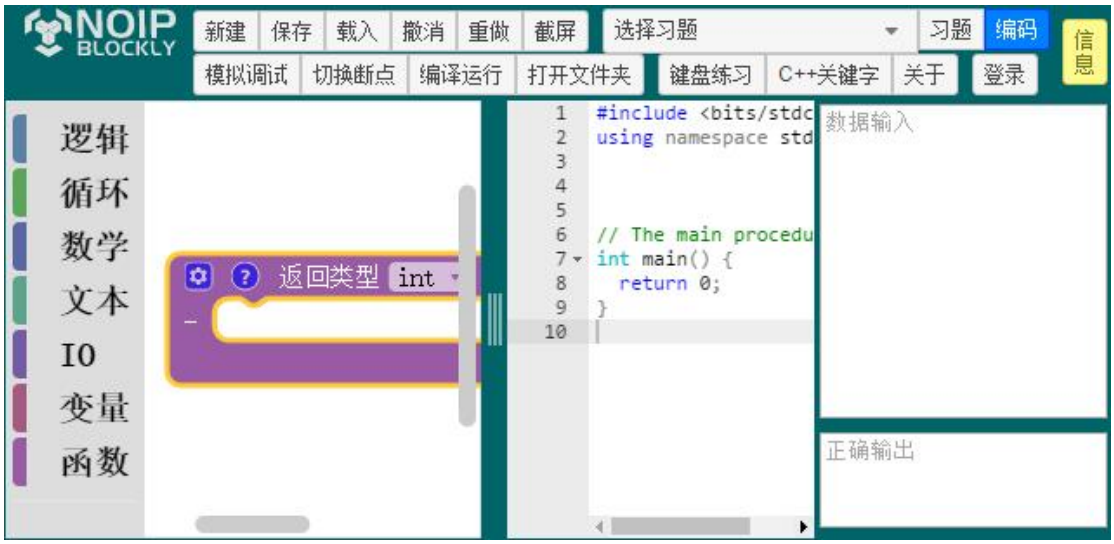
Noip 竞赛内容主要是 C++编程算法和数据结构，学习难度较大。Noip Blocky 编辑器专为参加 Noip 竞赛学习 C++而开发。本程序建立在谷歌 Blockly 之上，可以拖动块生成 C++代码。 这不仅让学生更容易理解 C++代码的运行过程，更是降低了 C++的学习难度。 图形化的编程方式，让学员可以形象化的分析实现算法，建立更高级的思维方式。并且和小学及初中信息技术课中的 Scratch 教学内容相对应，方便让学生把程序设计知识从 Scratch 迁移到 C++。

3. Noip Blocky 编辑器的安装

软件下载：http://sfzd5.com/noipblockly/web/
软件依赖：Noip Blocky 编辑器使用 C#开发，正常运行本软件需要安装下面的程序：

- Dev C++

4. Noip Blocky 编辑器界面



如上图所示，本编辑器是一个集成式的 Noip 学习软件，学生可以在这一个软件中完在读题、编写程序、编译运行全过程。

主要内容有：习题列表、Blockly 代码编辑、C++代码生成、输入输出数据、编译运行、结果判别等内容。

5. 常用块种类

代码生成块大致可以分成三种：

- 简单块。为顺序执行中的一步。

- 包含块。内部可包含其它块。

- 左接块。可以连接到块的接口上，向其它块提供数据。

块接口可分为内嵌接口和右接口。

6. 模拟调试运行代码

我们以完成“Hello World [1]”习题为便来学习软件的使用过程。

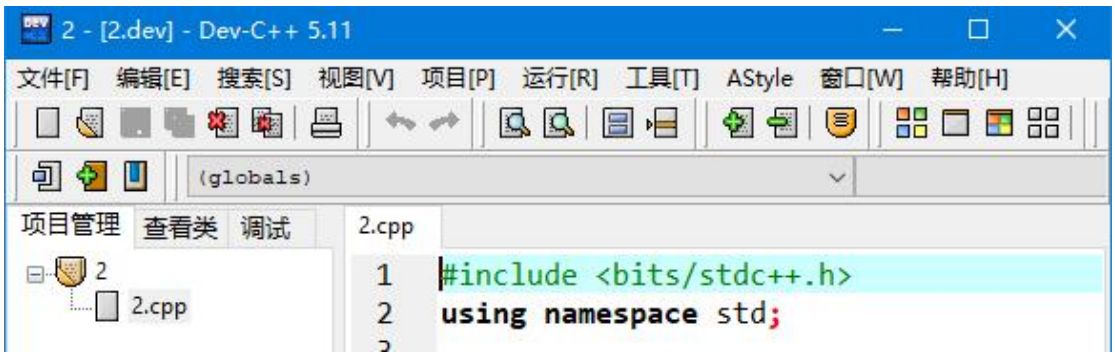
- 在左侧习题列表中点选“Hello World [1]”习题，此时在右侧“习题”选项卡中可以看到题目信息，我们可以阅读题目。
- 将题目的正确答案填在右侧输出框中。
- 点开“编码”选项卡，在 IO 功能组中拖动输出块到 main 函数中，并修改文字为 Hello world!。结果如下图：
- 

- 点击菜单“模拟调试”

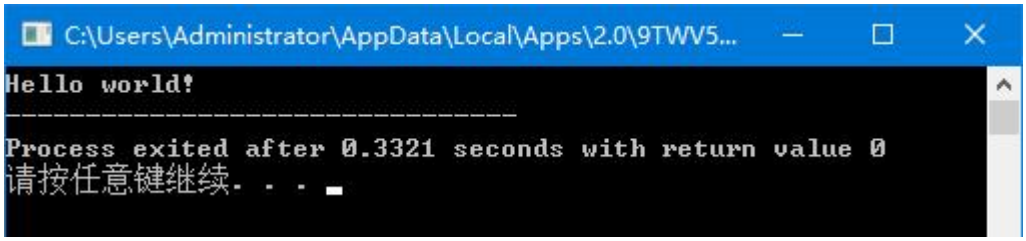
- 点击菜单“运行”
- 看运行结果是否正确

7. 在 Dev C++中编译运行

- 点击菜单“编译运行”



- 点击  “编译运行”
- 查看运行结果



8. 键盘练习

编程离不开键盘，所以必须学会正确高效的使用键盘。
打字时，应该将左右手的大拇指放在空格键上，其余手指分别放在 ASDF 和 JKL; 键上。打字时手指的分工图如下：

打字时注意事项

- 每打完字后，要迅速将手指移回初始位置。
- 手指不要弹的太高，也不要死按在键盘上，要保持灵活。

9. 作业

- 完成习题【Hello World [1]】
- “C++关键字”打字速度达到每分钟 60 个字符，并保持正确率 100%。

第一章 语法基础

C++是一种计算机高级程序设计语言，由 C 语言扩展升级而产生，最早于 1979 年由本贾尼·斯特劳斯特卢普在 AT&T 贝尔工作室研发。

C++既可以进行 C 语言的过程化程序设计，又可以进行以抽象数据类型为特点的基于对象的程序设计，还可以进行以继承和多态为特点的面向对象的程序设计。C++擅长面向对象程序设计的同时，还可以进行基于过程的程序设计。

C++拥有计算机运行的实用性特征，同时还致力于提高大规模程序的编程质量与程序设计语言的问题描述能力。

本章节是 C++的入门章节，主要学习各 C++的基础语法。所谓 C++语法就是程序代码的基本规则。

本章的具体内容有：

1、输入输出

2、变量与算术运算

3、条件与分支

4、while 循环

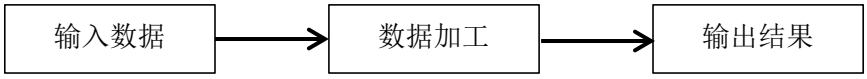
5、for 循环

6、多重循环及循环应用

1.1 C++输入与输出

1. 程序的工作过程

程序即按一定的顺序组织命令，来完成特定的工作的代码集合。大致运行过程如下：



C++程序从 main 函数开始执行。

2. 数据输入与存储

如上图：Noip Blockly 同 Scratch 一样，是通过拖动功能块来生成代码的。在“IO”功能中有“输出” 、“输入到”功能块。

输入数据时，必须为数据准备好存储的空间，即定义一个变量。

定义变量即为内存中某一块空间起一个名字，定义后即可以通过这一个名字来读取或更改这块空间内的数据。

输入数据到变量的代码如下：

生成 C++代码如下：

```cpp
#include <bits/stdc++.h>
using namespace std;

int a;

int main() {
  cin>>a;
  return 0;
}
```

编写过程为：

- 创建全局变量 a
- 添加“输入到”块到“main”函数中。（C++程序从 main 函数开始执行）。
- 添加变量“a”到“输入到”块后。

完成上面的操作即可生成右侧 C++代码。

提示：添加功能块时，需要对上缺口。

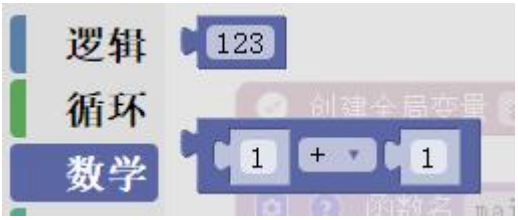
3. 输出

使用 

可以将数据输出到屏幕或文件。可以输出变量或文字数字等。

4. 数学运算

在“数学”功能中提供了许多数学运算功能。如下图



数学运算式可以嵌套执行，执行顺序为由内到外。示例如下：

5. 例题：A + B Problem

题目描述

输入 a 和 b，输出 $a+b$ 的结果。

输入格式

一行两个正整数 a 和 b。

输出格式

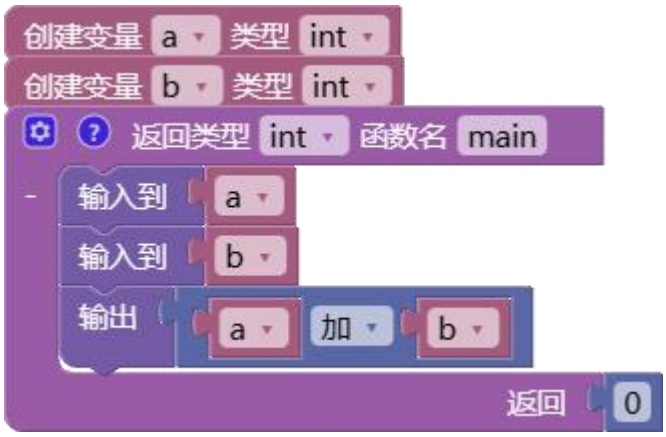
一行一个正整数 $a+b$。

如上图所示，习题大多包含三个内容：题目描述、输入格式、输出格式。一定要认真读题，分析清楚题意，才能写出下正确的程序。

习题分析如下：

程序要输入两个数据，所以要定义两个变量，并通过代码把数据输入到这两个变量中。再计算出这两个数据的和，并输出出来。

解题代码如下图：



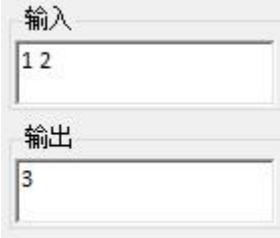
生成 C++代码如下：

```
#include <bits/stdc++.h>
using namespace std;

int a;
int b;

int main() {
```

```
    cin>>a;
    cin>>b;
    cout<<(a+b);
    return 0;
}
```

6. 准备输入数据和输出结果

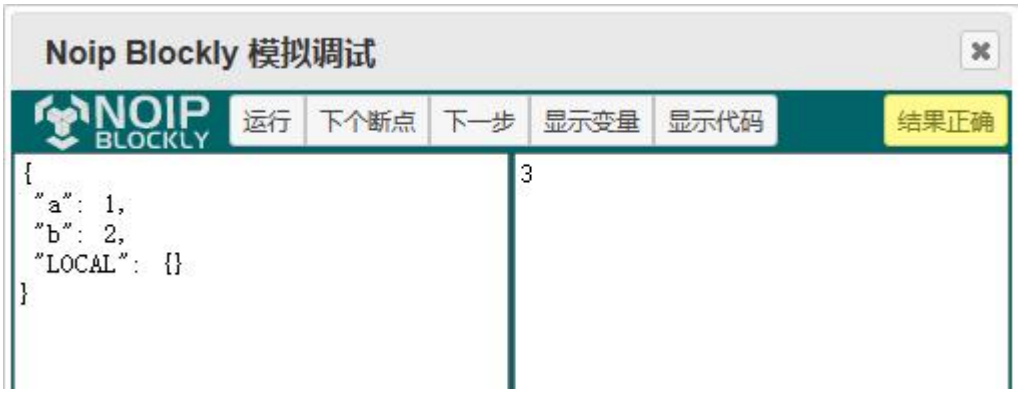
将程序中要用到的**输入数据**和**正确结果**填写到程序中，如下图：



输入的两个数据中间用空格隔开。输出用来判断程序是否正确执行。

7. 运行代码

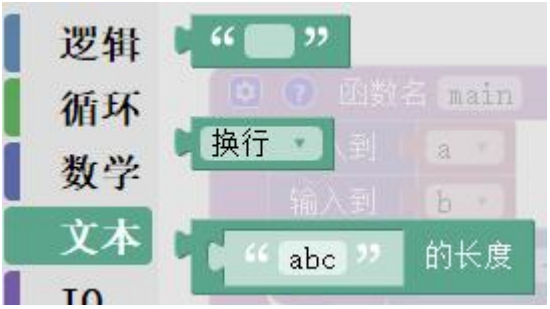
前面我们学习了两种运行代码的方式：模拟调试、编译运行。可以分别按这两种方法运行我们写的代码。并查看运行结果是否正确。
代码如下图：



8. 习题

完成【喊出我的名字!】、【第二个数】、【A+B Problem[1]】。

习题提示：
【喊出我的名字!】
需要用到“文本”功能中的“换行”功能块。


逻辑
循环
数学
文本
函数名 main
换行
a
输入到
b
abc
的长度

1.2 变量与算术运算

变量是程序的基本组成单位，变量是内存中的一个数据存储空间。你可以把变量看做是一个房间的门牌号，通过门牌号我们可以找到房间，从而通过变量名可以访问到变量(值)。

1. 定义变量

上面是创建全局变量的块，右侧为生成的代码。上图中定义了一个名字叫 a 的变量，a 变量用来存储 int 类型的数据。
变量分为全局变量和局部变量。全局变量可以在程序任何位置使用；局部变量只能在包含块及其子块中使用，。

2. 数据类型

数据类型即存储在内存中的内容类型，有以下几类：

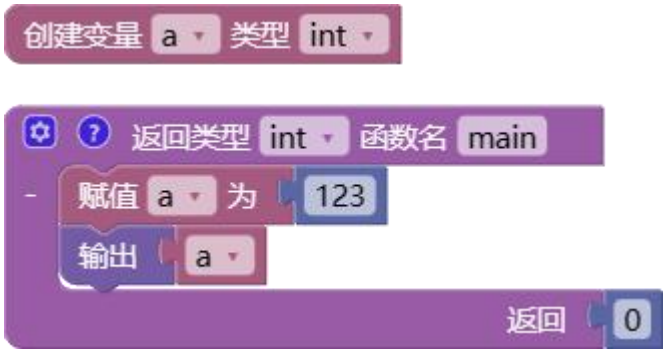
- 整型 int、长整型 long long、短整型 short 变量可以存储整数，但可存储的最大最小值不同。
- 浮点型 float、双浮点型 double 变量可以存储小数，但可存储的最大最小值不同。
- 布尔型 bool 变量可以存储真或假（true 或 false）两种值。
- 字符型 char 变量可以存储一个字符，如：a b c A 1 2 3 等
- 字符串 string 变量可以存储一串字符，如：”I am a good boy.”

除 string 外上面的变量类型均为基础数据类型，基础数据类型有固定的存储长度。
string 严格来说并不能算一种数据类型，它是一个功能类。string 提供了存储和操作一串字符的各种功能。

3. 变量的赋值和读取

在定义变量 a 后，可以在“变量”功能中看到变量赋值和读取功能块。

如下为赋值和读取的示例：



生成的 C++代码如下：

```
#include <bits/stdc++.h>
using namespace std;

int a;

int main() {
  a = 123;
  cout<<a;
  return 0;
}
```

上图中生成的 C++代码：

```
  a = 123;
```

上面这一行代码是 C++的赋值语句，这和数学中的等式不同。这个赋值语句的意思是**将 123 存储到变量 a 中**。

4. 算术运算

C++常用算术运算主要有：加、减、乘、除、模（取余）。

C++的算术运算中有些与我们学过的数学不太一样的地方，大家要注意：

- 整数除法。

要注意，整数相除时，如 10 除以 4，可不是 2.5，而是 2。int 类型，不能存小数的。如果将小数赋值给了 int 变量，小数点后的部分被抛弃。如果要计算小数，那就必须使用浮点类型 float 或 double 。

- 不同种类型数学运算会自动转换为复杂的类型。如果想把一个整型转化为浮点型，只需要让它乘上 1.0 即可。如 5*0.5 结果为浮点型。C++中类型复杂由小到大顺序如下：

short < int < long < float < double

- char 也可以参与数学运算，运算结果会自动转变为 int 类型。
- 取余运算即计算余数。例：5%3 的值为 2

数学运算的结果可赋值给变量或直接输出。下图将变量 a 与 123 的和赋值给了变量 a 。

除了这些基础数学运算，C++数据学运算库中也有许多更加高级的运算，如：乘方、开方、绝对值、三角函数等。

5. 字符编码

电脑中只能保存数字，为了保存图像、声音、文字等数据，就要对它们进行编码。

字符型（char）变量中存放的就是字符，其本质是一个数字。下表为部分字符 ascii 码表

编码	符号		编码	符号		编码	符号		编码	符号		编码	符号
48	0		65	A		78	N		97	a		110	n
49	1		66	B		79	O		98	b		111	o
50	2		67	C		80	P		99	c		112	p
51	3		68	D		81	Q		100	d		113	q
52	4		69	E		82	R		101	e		114	r
53	5		70	F		83	S		102	f		115	s
54	6		71	G		84	T		103	g		116	t
55	7		72	H		85	U		104	h		117	u
56	8		73	I		86	V		105	i		118	v
57	9		74	J		87	W		106	j		119	w

			75	K		88	X		107	k		120	x
			76	L		89	Y		108	l		121	y
			77	M		90	Z		109	m		122	z

所以字符也可以参与数学运算。

6. 运算优先级

C++的数学运算优先级也是先乘除后加减，有小括号，先算小括号。

例：

先算最内部的 a - b 的差，再算 26 加上这个差。

7. 数据类型转换

简单的数据类型转换有两种方法：

1. 赋值转换

当不同基础类型的变量相互赋值时，会发生类型转换。

当由小数类型转为整数类型时，小数部分会被舍弃（注意不是四舍五入）。

2. 强制转换

在变量前添加 括号+类型 即可。示例如下：

```
Double a = 12.5;
Cout<<(int)a;
```

经过强制转换后，上面代码最后输出：12 ，小数点后的部分被舍去。

8. 习题练习

完成“变量与数据类型”练习题

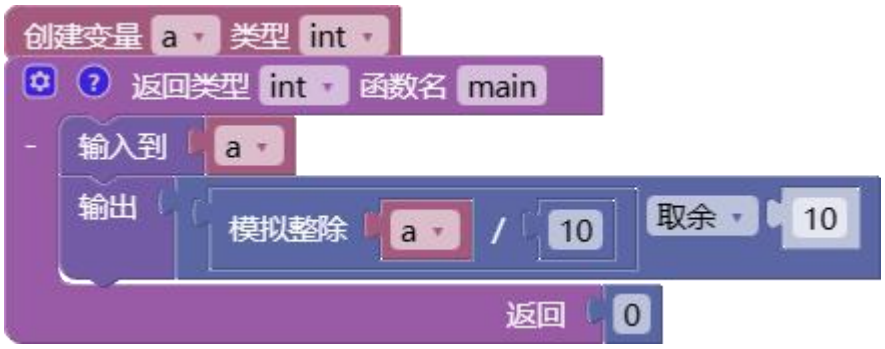
部分习题提示：

- 【整数的个位】

运用求余运算。例：14%10，求出来的余是 4

- 【整数的十位】

使用 c++整数相除时得数仍然是整数特点，以及求余运算来完成本题。



- 【保留三位小数】

小数需要使用浮点数据类型，并且设置输出的小数位。

- 【浮点除法】

要有小数，必须使用浮点数据类型。

- 【字符的 ASCII 码】

需要把字符型转为数值。

- 【大小写转换】

参考字符编码，需要用到数学运算和数据类型转换。

1.3 条件与分支

电脑都是通过按特定顺序执行一系列操作来解决问题，编写这一操作过程就是编程。程序中代码的运行的不同过程可以分为三类，即代码的三种控制结构：顺序结构、选择结构和循环结构。本节讲选择结构。

1. 关系运算

关系运算的结果只有两种：即真或假。选择结构和循环结构中需要根据一定的关系运算结果来选择执行分支或判断是否要重复执行。
主要有下面几种关系运算

C++关系运算符	举例	含义
==	x==y	x 等于 y
!=	x!=y	x 不等于 y
<	x<y	x 小于 y
<=	x<=y	x 小于或等于 y
>	x>y	x 大于 y
>=	x>=y	x 大于或等于 y

关系运算的结果是布尔型 bool，只能为真 true 或假 false。
例：
5>3 的结果为： true
12%2==1 的结果为：false

2. 逻辑运算

逻辑运算是对多个关系运算的结果进行组合判断。逻辑运算的结果也只有两种：即真或假。
常用逻辑运算有三种：与、或、非。见下图

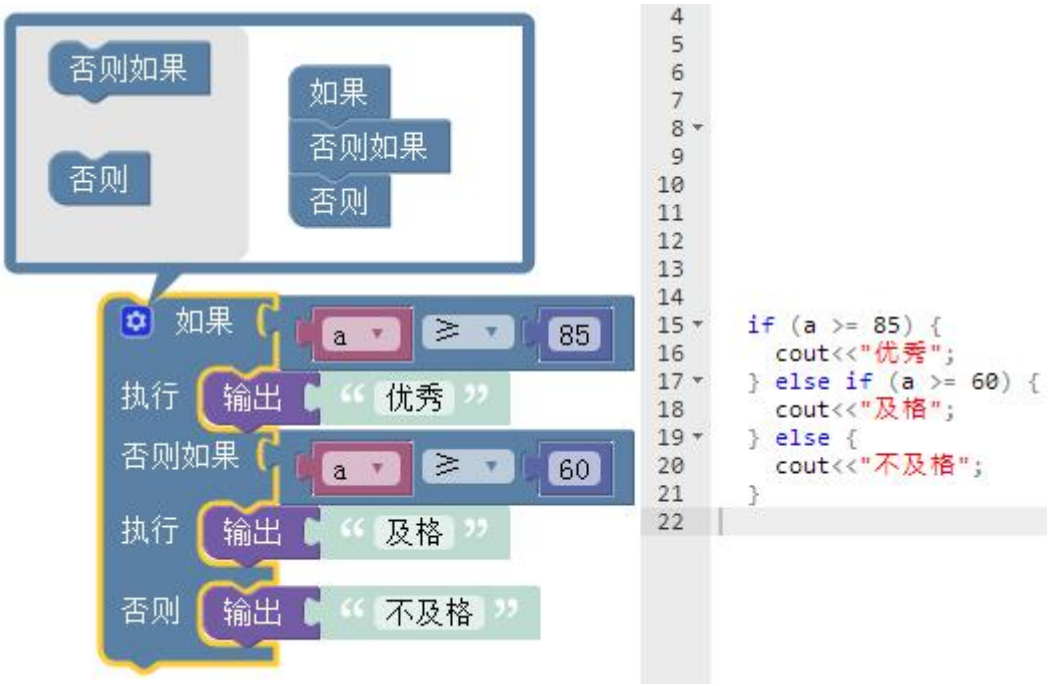
运算符	举例	含义
&&	x&&y	均为真结果为真，否则为假
||	x||y	一个为真结果为真，否则为假
!	!x	真变假，假变真

上表中 x、y 为布尔变量。
逻辑运算可以嵌套，运算顺序为从内到外，从左到右。

例：
5>3 || 12%2==1 的结果为：true
5>3 && 12%2==1 的结果为：false

3. if/else 选择结构

选择结构是在程序运行时，依据关系运算或逻辑运算的结果，选择不同的程序分支，运行不同的程序代码。
如下为判断考试成绩等次的代码：



如上图，程序对成绩进行比较，根据逻辑运算结果，会输出不同的文字。

4. 例题【判断是否被 3 和 5 整除】

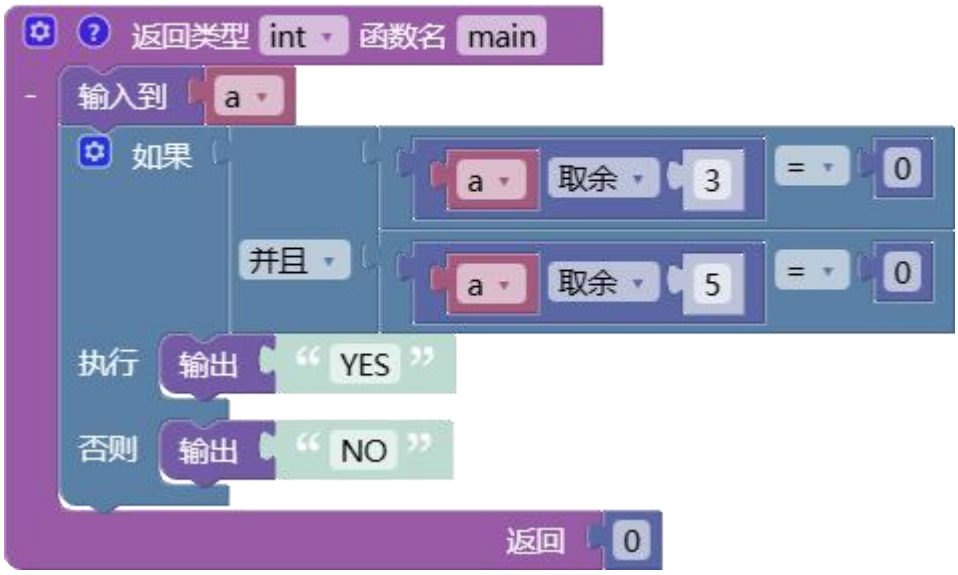
题目描述
判断一个数 n 能否同时被 3 和 5 整除
输入格式
输入一行，包含一个整数 n 。（ -1,000,000 < n < 1,000,000 ）
输出格式
输出一行，如果能同时被 3 和 5 整除输出 YES ，否则输出 NO 。
样例
输入样例 1
15
输出样例 1
YES
输入样例 1
12

输出样例 1

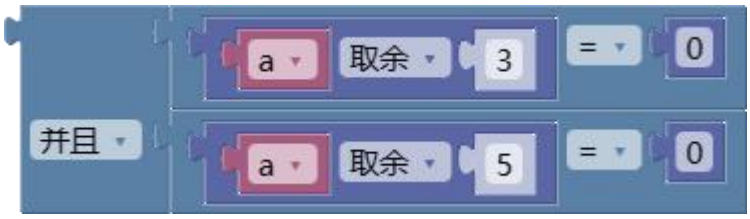
NO

这个题中需要同时判断一个数和 3、5 的整除情况，所以需要同时用到关系运算和逻辑运算。

代码如下：



逻辑运算组合的样子如下：



上面将两个关系运算的结果又进行了“并且”逻辑运算。

生成的 C++代码如下：

```cpp
#include <bits/stdc++.h>
using namespace std;

int a;

int main() {
  cin>>a;
  if (((a%3) == 0) && ((a%5) == 0)) {
    cout<<"YES";
  } else {
    cout<<"NO";
  }
  return 0;
}
```

5. 【被 3，5，7 整除】

题目描述

给了你一个整数，请判断它能否被 3,5,7 整除，并输出以下信息：

能同时被 3, 5, 7 整除（直接输出 3 5 7 ，每个数中间一个空格）；

只能被其中两个数整除（输出两个数，小的在前，大的在后。例如：3 5 或者 3 7 或者 5 7，中间用空格分隔）；

只能被其中一个数整除（输出这个除数）；

不能被任何数整除，输出 NO 。

输入格式

输入一行，包括一个整数

输出格式

输出一行，按照描述要求给出整数被 3,5,7 整除的情况。

样例

输入样例 1

105

输出样例 1

3 5 7

输入样例 2

30

输出样例 2

3 5

这个题需要对多种情况进行完整的判断。分析可知共需要判断 7 回，若 7 次判断都不满足，则不能被任何一个数整除，则输出 NO。

代码如下：

C++代码如下：

```
1. #include <bits/stdc++.h>
2. using namespace std;
3.
```

```
int a;
bool k;

int main() {
  cin>>a;
  k = true;
  if ((a%3) == 0) {
    cout<<"3";
    k = false;
  }
  if ((a%5) == 0) {
    if (!k) {
      cout<<" ";
    }
    cout<<"5";
    k = false;
  }
  if ((a%7) == 0) {
    if (!k) {
      cout<<" ";
    }
    cout<<"7";
    k = false;
  }
  if (k) {
    cout<<"NO";
  }
  return 0;
}
```

6. 习题练习

完成“C++基础 L1 条件与分支”

部分习题提示：

- 【判断奇偶】

运用求余运算，判断数值除以 2 的余数是否为 0

- 【绝对值】

多运用我们学过的数学知识。比如：正负数的概念、如何将一个负数变为正数。

- 【判断奇偶 2】

字符本质上也是一个数字

- 【判断是否被 3 和 5 整除】

同时运用关系运算和逻辑运算

- 【被 3，5，7 整除】

逻辑较复杂，一定要考虑全面。

- 【判断闰年】

熟练运用关系运算和逻辑运算

- 【点与正方形】

画出图形，并分析正方形中点的坐标的特点，找出规律。

- 【最大值最小值】

熟练运用嵌套逻辑运算。理清逻辑顺序。

- 【三角形判断】

利用三角形性质：两边和大于第三边

- 【工作时间】

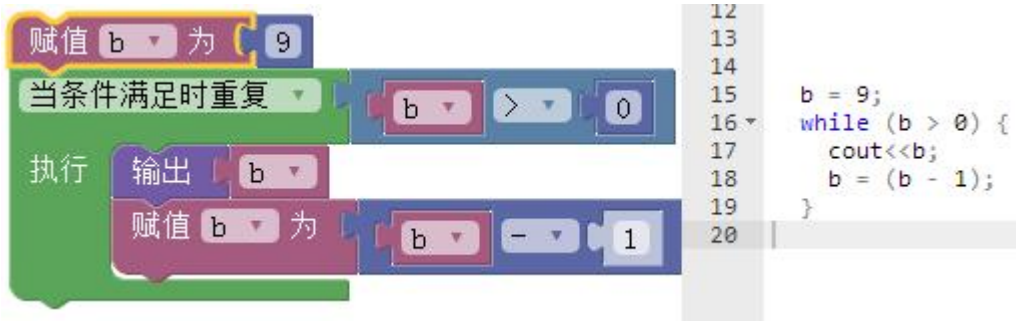
方法很多，合理即可

1.4 while 循环

电脑都是通过按特定顺序执行一系列操作来解决问题，编写这一操作过程就是编程。程序中代码运行的不同过程可以分为三类，即代码的三种控制结构：顺序结构、选择结构和循环结构。本节讲循环结构中的 while 循环。

while 循环为条件循环，是根据逻辑运算的结果重复执行块中的代码。

1. while 循环举例讲解



如上图中循环执行的条件是“b>0”，即当“b>0”时循环执行中间的两个语句，即输出和赋值语句。每次循环 b 的值就会减少 1，b 不断减小，直到为 0 时循环停止。

以上循环共执行了 9 次。

最终输出结果为：987654321

注意：循环一定要有停止的条件，不能一直循环。

2. 【自然数的和】

题目描述

输入一个自然数 N ，求 1 到 N 的自然数之和 。

请使用循环结构来完成本题

输入格式

输入一行，包含一个整数 N

输出格式

输出一行，包含一个整数。

输入样例

100

输出样例

5050

习题分析：

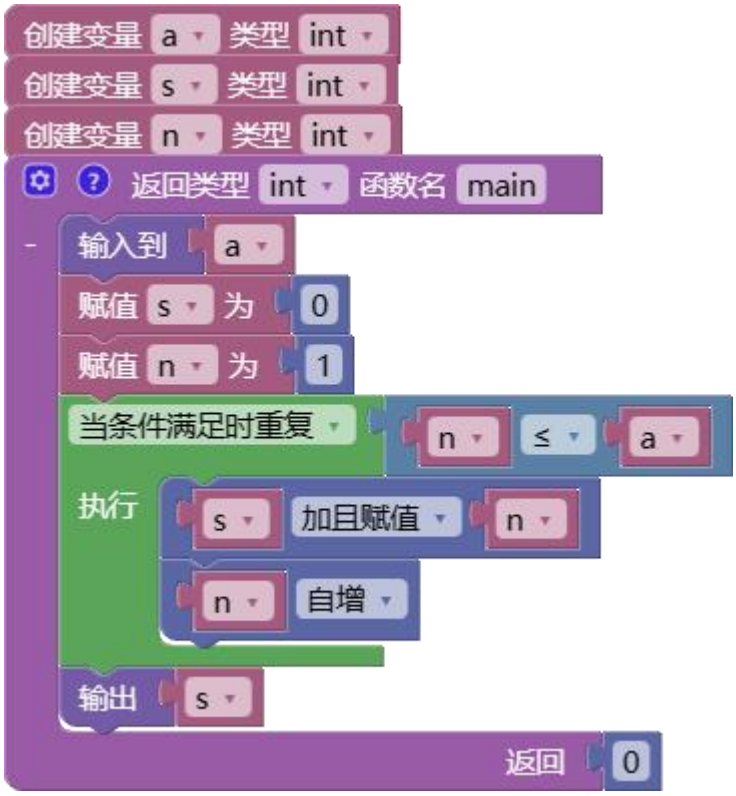
可以通过一个 while 循环，产生从 1 到 N 所有的数。然后将这些数**累加**起来就可以了。

其中累加的运算如下图：

与下面代码等效：

注意：在累加之前，还要对 s 进行初始化赋值为 0。

完整代码如下：



生成的 C++代码如下:

```cpp
#include <bits/stdc++.h>
using namespace std;

int a;
int s;
int n;

int main() {
  cin>>a;
  s = 0;
  n = 1;
  while (n <= a) {
    s += n;
    n++;
  }
  cout<<s;
  return 0;
}
```

上图中用到了“自增”运算，即在变量自身减去 1 。类似的还有“自减”运算。

3. 【一五一十】

题目描述

藤藤给了你 k 个正整数，其中每个数都是大于等于 1 ，小于等于 10 的数。写程序计算给定的 k 个正整数中， 1 、5 和 10 出现的次数。

输入格式

输入有两行：第一行包含一个正整数 k ，第二行包含 k 个正整数，每两个正整数用一个空格分开。

输出格式

输出有三行，第一行为 1 出现的次数，第二行为 5 出现的次数，第三行为 10 出现的次数。

样例

输入样例

5
1 5 8 10 5

输出样例

1
2
1

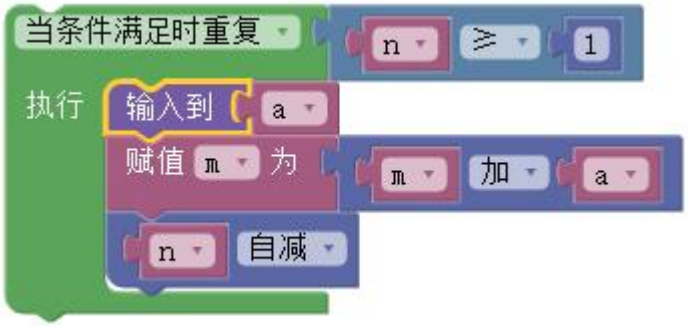
习题分析：
这是一个统计计数题。和我们数数一样，每碰到一个符合要求的，计数变量就加上 1.
上图中用到了“自减”运算，即在变量自身减去 1 。类似的还有“自增”运算。

4. 习题练习

完成“C++基础 L1 while 循环”
习题提示：

- 【求和】

在 n 次循环中，输入 n 个数字。在循环中写输入的代码。



- 【藤藤学数列】

观察可知，当数字为奇数时要加，当数字为偶数时要减。

- 【藤藤学数列 2】

变量类型要用浮点型

- 【一五一十】

要定义计数的变量，初始值为 0。碰到符合条件的情况就让相关的计数变量加 1

- 【幂次计算】

只需要把累加改为累乘即可。注意初始值的设置。

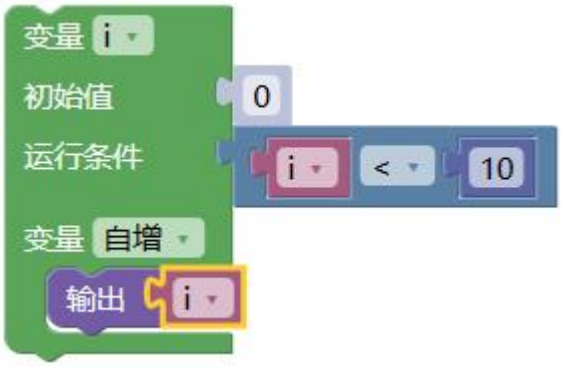
- 【平均年龄】

注意使用浮点型变量和保留小数位。

1.5 For 循环

for 循环为指定变量条件循环，是根据指定变量值的变化，进行逻辑运算，依据逻辑运算结果重复执行块中的代码。

1. for 循环举例讲解



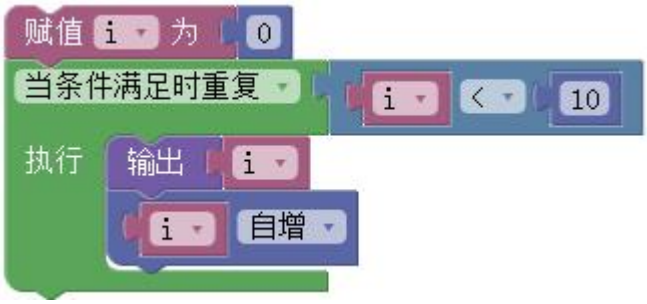
生成 C++代码如下：

```
for (int i = 0; i < 10; i++) {
    cout<<i;
}
```

如上图中循环指定变量为 i ，初始值为 0 ，执行的条件是“i<10”，即当“i<10”时循环执行中间的两个语句。每次循环 i 的值就会增加 1，直到为 10 时循环停止。

最终输出结果为：0123456789

上面的 for 循环与下面的 while 循环结果是一样的。



由上图可看到，for 循环比 while 循环少了两个语句。由此可见，for 循环写起来比 while 简单，但 while 循环更灵活。

2. 跳出循环或继续下一轮循环

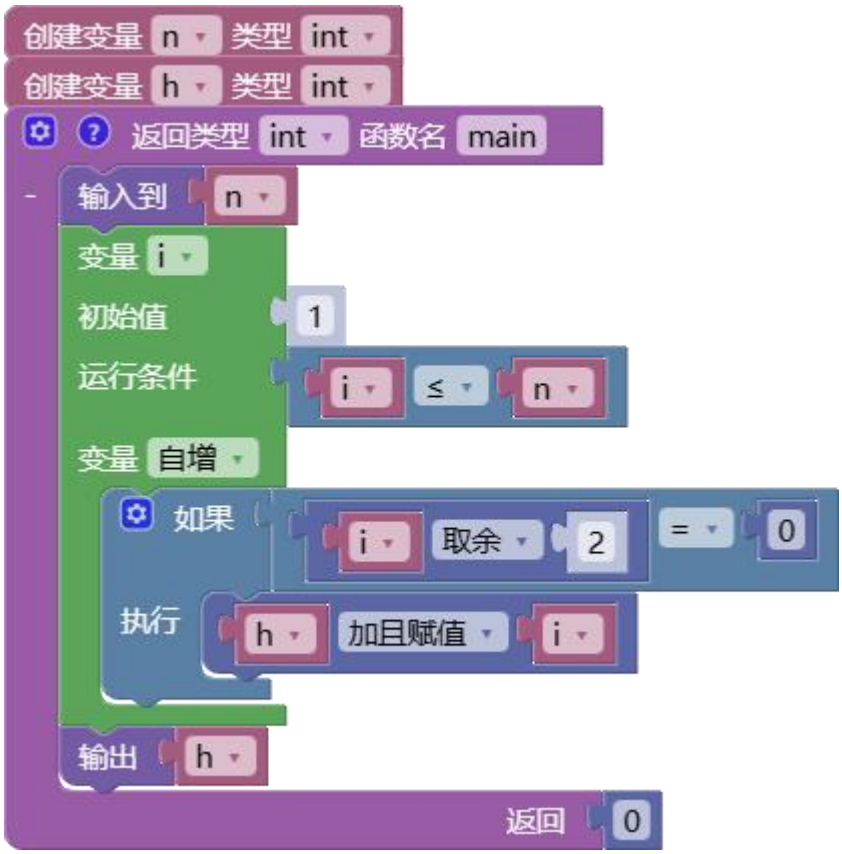
如上图，在循环中，可以根据一定的条件，直接跳出当前循环，或直接继续下一轮循环。对

应的 C++代码为：跳出循环：break ；继续下一轮循环：continue 。

3. 【1 到 n 所有偶数的和】

题目描述
输入，求到中所有偶数的和。
输入格式
输入为一行，包括一个正整数。
输出格式
输出 1 到 n 中所有偶数的和，保证和在的范围内
样例
样例输入
4
样例输出
6

代码如下：



生成 C++如下：

```
#include <bits/stdc++.h>
using namespace std;

int n;
int h;
```

```
6.
7.  int main() {
8.    cin>>n;
9.    for (int i = 1; i <= n; i++) {
10.     if ((i%2) == 0) {
11.       h += i;
12.     }
13.   }
14.   cout<<h;
15.   return 0;
16. }
```

习题分析：上面的代码中，用循环产生了 1 到 n 的所有数。经过判断，如果产生的数是偶数，就将这个数累加到一个变量中。
上面代码中又用到了一个新块：加且赋值，它的功能是在变量自身的基础上再加一个量。类似的还有：减且赋值、乘且赋值、除且赋值。

4. 编程技巧小结

- 可利用循环产生需要的数据。
- 利用“加且赋值”累加数据。

5. 习题练习

完成“C++基础 L1 for 循环”

- 【幸运数字 7】
- 【1 到 n 所有奇数的和】
- 【1 到 n 所有偶数的和】
- 【m 到 n 的和】
- 【求和】
- 【藤藤问西湖龙井】

1.6 多重循环

多重循环指循环中嵌套有循环。

1. 例题讲解

例题 1.【打印练习 2】

题目描述
输入一个整数 n ，输出 n 行 n 列的矩形图案。
输入格式
一行，包含一个整数 n 。
输出格式
n 行 n 列的矩形图案
样例
输入样例
4
输出样例
****
****
****
****

题目内容较好理解。分析如下 ：
最终的输出结果是一个长方形的字符块。输出的时候逐行输出，这就需要一个逐行输出的循环；每一行的每个字符需要逐个输出，这又需要一个循环。这就需要两个循环，外层按行循环，内层按字符循环，每次只输出一个字符。
代码如下：

C++题解代码

```
#include <bits/stdc++.h>
using namespace std;

int n;


// The main procedure
int main() {
  cin>>n;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= n; j++) {
```

```
12.         cout<<"*";
13.       }
14.       cout<<'\n';
15.     }
16.     return 0;
17. }
18.
```

Blockly 题解代码图片

例题 2. 【双重循环基础】

题目描述

给出一个 n，打印出 n 行，
第一行一个数 1，
第二行两个数 1 2，...
第 n 行有 n 个数 1 2 3 4.....n

每行中的数字之间空格隔开
输入格式
一个正整数 n
输出格式
如问题描述
样例
样例输入 1
3
样例输出 1
1
1 2
1 2 3

题目内容较好理解。分析如下 ：
第一个问题我们应想到输入的是几就应该输出几行，这需要一个循环来完成；
第二个问题，比如我们要输出第 m 行，第 m 行的内容应该是 1——m 的数字，数字间有空格隔开，这里我们还需要一个 for 循环产生这些数字。
这样我们就需要两层循环。并且内部的循环次数 m 即行号，即外部的循环的变量值。
代码如下：

C++题解代码

```cpp
#include <bits/stdc++.h>
using namespace std;

int n;


// The main procedure
int main() {
  cin>>n;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= i; j++) {
      cout<<j;
      if (i > j) {
        cout<<" ";
      }
    }
    cout<<'\n';
  }
  return 0;
}

```

Blockly 题解代码图片

如图，在第二层循环的执行条件中使用了外层循环的变量值。注意输出的格式。

例题 3. 【字符三角形】

题目描述
给定一个字符，用它构造一个底边长 5 个字符，高 3 个字符的等腰字符三角形。
输入格式
输入只有一行， 包含一个字符。
输出格式
该字符构成的等腰三角形，底边长 5 个字符，高 3 个字符。
样例
样例输入 1

```
*
```

样例输出 1

```
  *
 ***
```

```
*****
```

题目分析如下 ：

本题要输出一个三角形的字符块，根据上面两个题的经验，我们也可以按逐行逐列的输出。但本题明显比上面两题要复杂，需要考虑的问题也越多。主要有下面和个问题：

1. 本题的*号，从上到下，每行增加 2 个*号。这需要在循环内体现。
2. 除了要输出*号，还要在左侧输出空格，才能让*到恰当的位置，形成三角形。
3. 每行的空格数等于 3 减去当前行号。第 1 行的空格数为 3-1 ，2 个。

明白上面这些问题后，可以外部需要一个行号的循环；外部循环里面需要两循环，分别用输出空格和*号。

代码如下：

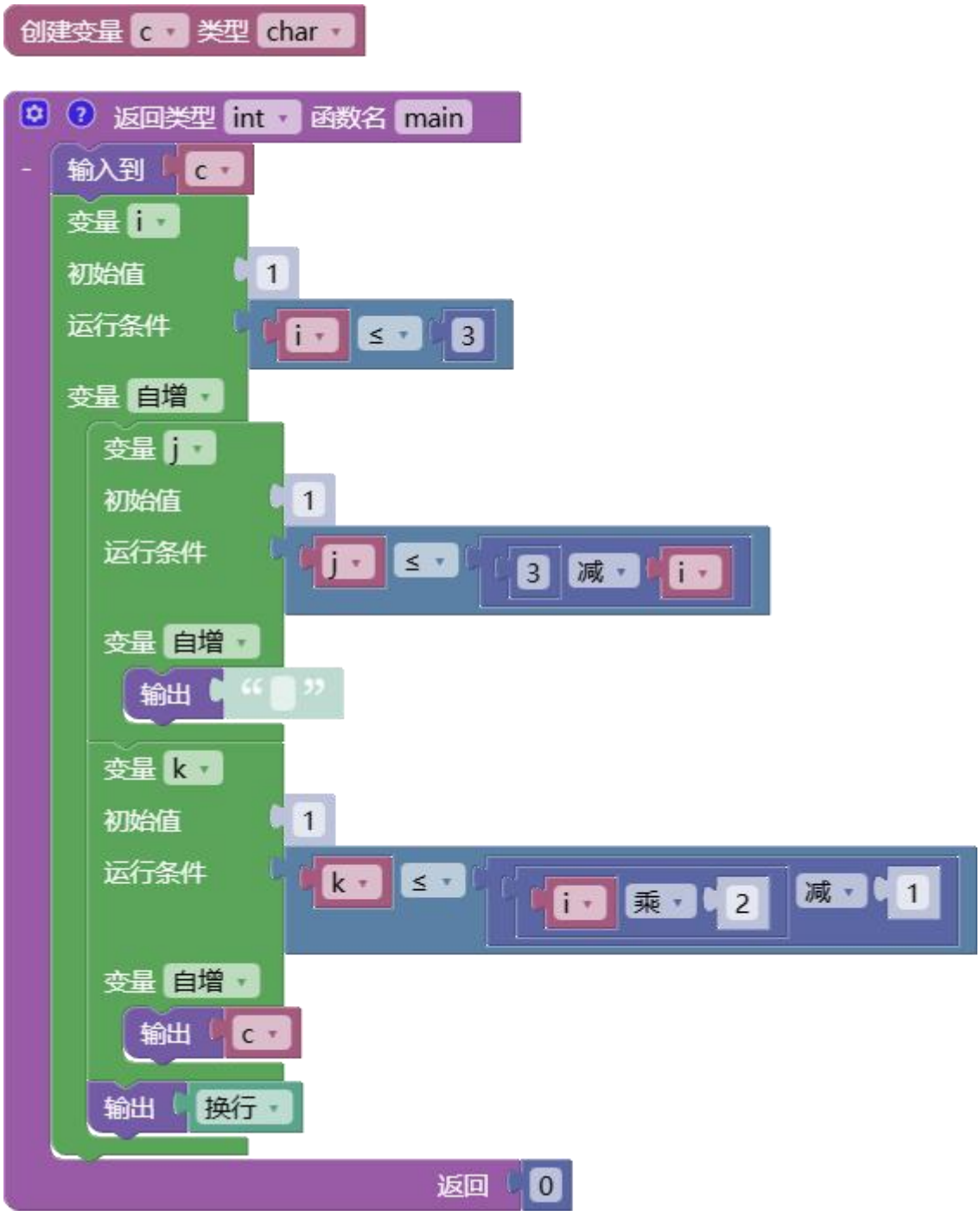
C++题解代码

```
#include <bits/stdc++.h>
using namespace std;

char c;


// The main procedure
int main() {
  cin>>c;
  for (int i = 1; i <= 3; i++) {
    for (int j = 1; j <= (3-i); j++) {
      cout<<" ";
    }
    for (int k = 1; k <= ((i*2)-1); k++) {
      cout<<c;
    }
    cout<<'\n';
  }
  return 0;
}

```

编程技巧小结

- 当处理有多行多列的数据时，可以用双重循环，逐行逐列的处理数据。
- 多重循环中，一定要明白每层循环的作用和意义。
- 可通过运算，动态得出循环的次数和顺序。
- 将复杂问题分解为简单的问题。

Blockly 题解代码图片



2. 习题练习

完成“C++基础 L1 多重循环”

1.7 循环应用（1）

循环是三种程序结构中的一种，是解决复杂问题必然要用的地方法。

1. 例题

例题 1.【确定一个整数的位数】

题目描述
输入一个不超过 10^9 的正整数，输出它的位数。例如 12735 的位数是 5 。请不要使用任何数学函数，只用四则运算和循环语句实现
输入格式
一个正整数 n， $n <= 10^9$ 。
输出格式
正整数 n 的位数
样例
输入样例
12735
输出样例
5

题目分析如下：
题目要求用数学运算的方式确定整数的位数，依据前面做过的取出第几位数字的例子，我们可以设计如下：

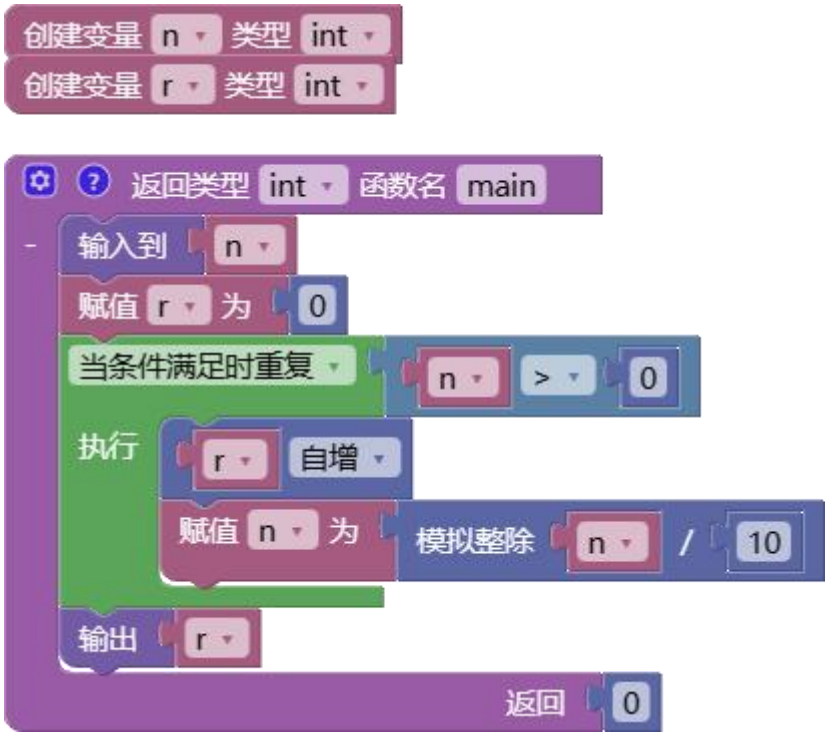
1. 让输入的数除以 10，这样这个数字必然少了一位；
2. 循环让该数除以 10，判断商是否大于 0，如果商大于 0，则重复以上步骤，否则结束循环；
3. 循环结束后，除以 10 的次数就是数字的位数。

C++题解代码

```cpp
#include <bits/stdc++.h>
using namespace std;

int n;
int r;


// The main procedure
int main() {
  cin>>n;
  r = 0;
```

```
  while (n > 0) {
    r++;
    n = (n/10);
  }
  cout<<r;
  return 0;
}

```

Blockly 题解代码图片



如上图：我们定义了两个全局变量。a 用来接收数据，c 用来计数。
main 函数中的执行过程：

① 输入数据到 a
② 初始化计数变量 c 为 0
③ 根据条件，当 “a>0”时循环
  1) 让 a 除以 10，并把商再赋值给 a
  2) 计数器变量 c 自增 1
④ 输出变量 c，即为位数

编程技巧小结

- 巧妙利用条件循环可以让代码具有普适性。
- 本题中展示了整除运算的妙用

例题 2. 【正整数能够整除几次 2？】

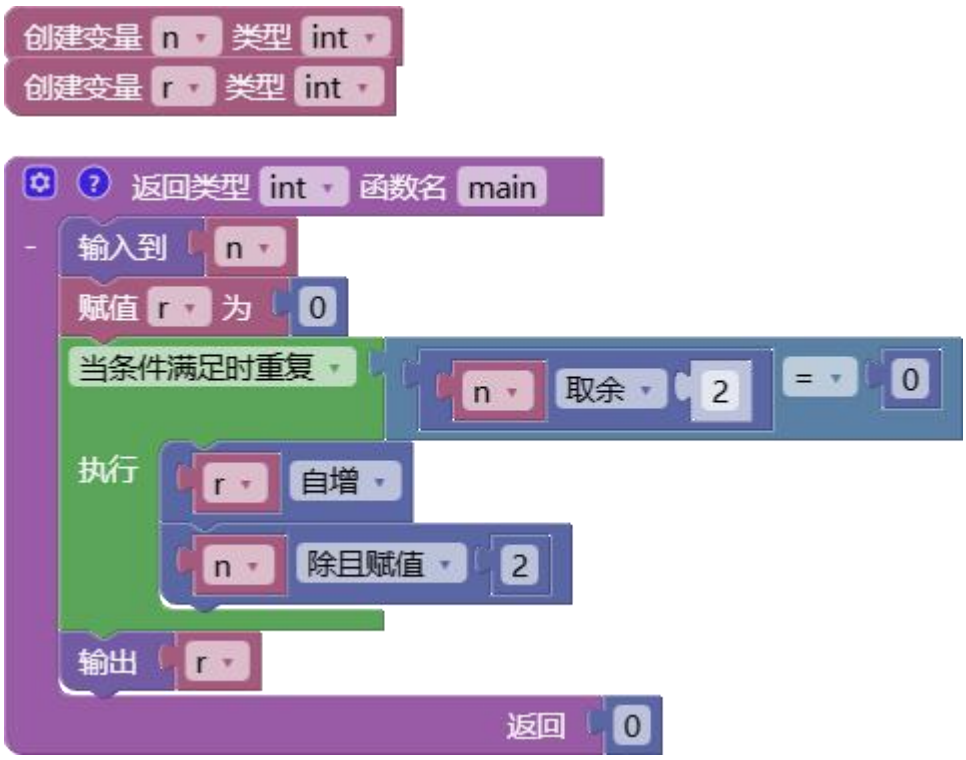
题目描述
请问一个正整数 n 能够整除几次 2？
比如：4 可以整除 2 次 2，100 可以整除 2 次 2，9 可以整除 0 次 2。
输入格式
从键盘读入一个正整数 n
输出格式
输出一个整数，代表 n 能够整除 2 的次数
样例
输入样例
8
输出样例
3

题目分析如下：
数学中学到，当两数相除余数为 0 时就是整除。我们可以设计如下：
1. 判断输入的数是不是能被 2 整除，即判断该数除 2 的余数是否为 0；
2. 循环判断该数是否能被 2 整除，如果能被整除，将该数除以 2，并循环重复上面的操作，否则结束循环；
3. 循环结束后，循环的次数就是数字的位数。

C++题解代码

```cpp
#include <bits/stdc++.h>
using namespace std;

int n;
int r;


// The main procedure
int main() {
  cin>>n;
  r = 0;
  while ((n%2) == 0) {
    r++;
    n /= 2;
  }
  cout<<r;
  return 0;
}

```

Blockly 题解代码图片



编程技巧小结

- 和上一题相同，利用 while 循环让代码具有普适性。
- 巧妙的运用了模运算和整除运算。

3. 习题练习

完成“C++基础 L1 循环应用（1）”

1.8 循环应用（2）

1. 例题

例题 1. 【水仙花数】

题目描述
春天是鲜花的季节，水仙花就是其中最迷人的代表，数学上有个水仙花数，他是这样定义的：“水仙花数”是指一个三位数，它的各位数字的立方和等于其本身比
如：$153 = 1^3 + 5^3 + 3^3$ 。
现在要求输出所有在 m 和 n 范围内的水仙花数。
输入格式
两个整数 m 和 n（100<=m<=n<=999）。
输出格式
要求输出所有在给定范围内的水仙花数，就是说，输出的水仙花数必须大于等于 m，并且小于等于 n，
如果有多个，则要求从小到大排列在一行内输出，之间用一个空格隔开；
如果给定的范围内不存在水仙花数，则输出 no。
样例
输入 #1
300 380
输出 #1
370 371

习题分析：
本题可以想到，可以分在下面两个部分来解决：
1. 我们可以用一个循环列出给出范围内的所有三位数；
2. 然后再计算各数位上的立方和并判断是否和原数相等。

第二步有些困难，但以前我们做过一些相关的问题，可以给我们一些提示。只需要在除 10 前先模 10，就可以逐一取出未位数。依次操作三回，就可以将数字的各数位取出来。将取出的数字的三次方累加一块，再与原数比较即可判断该数是不是水仙花数。

C++题解代码

```
#include <bits/stdc++.h>
using namespace std;

int n;
int g;
int s;
int b;
int m;
```

```
bool x;


// The main procedure
int main() {
  cin>>m;
  cin>>n;
  x = true;
  for (int i = m; i <= n; i++) {
    g = (i%10);
    s = ((i%100)/10);
    b = (i/100);
    g = ((g*g)*g);
    s = ((s*s)*s);
    b = ((b*b)*b);
    if ((g+(b+s)) == i) {
      if (x) {
        x = false;
      } else {
        cout<<" ";
      }
      cout<<i;
    }
  }
  if (x) {
    cout<<"no";
  }
  return 0;
}

```

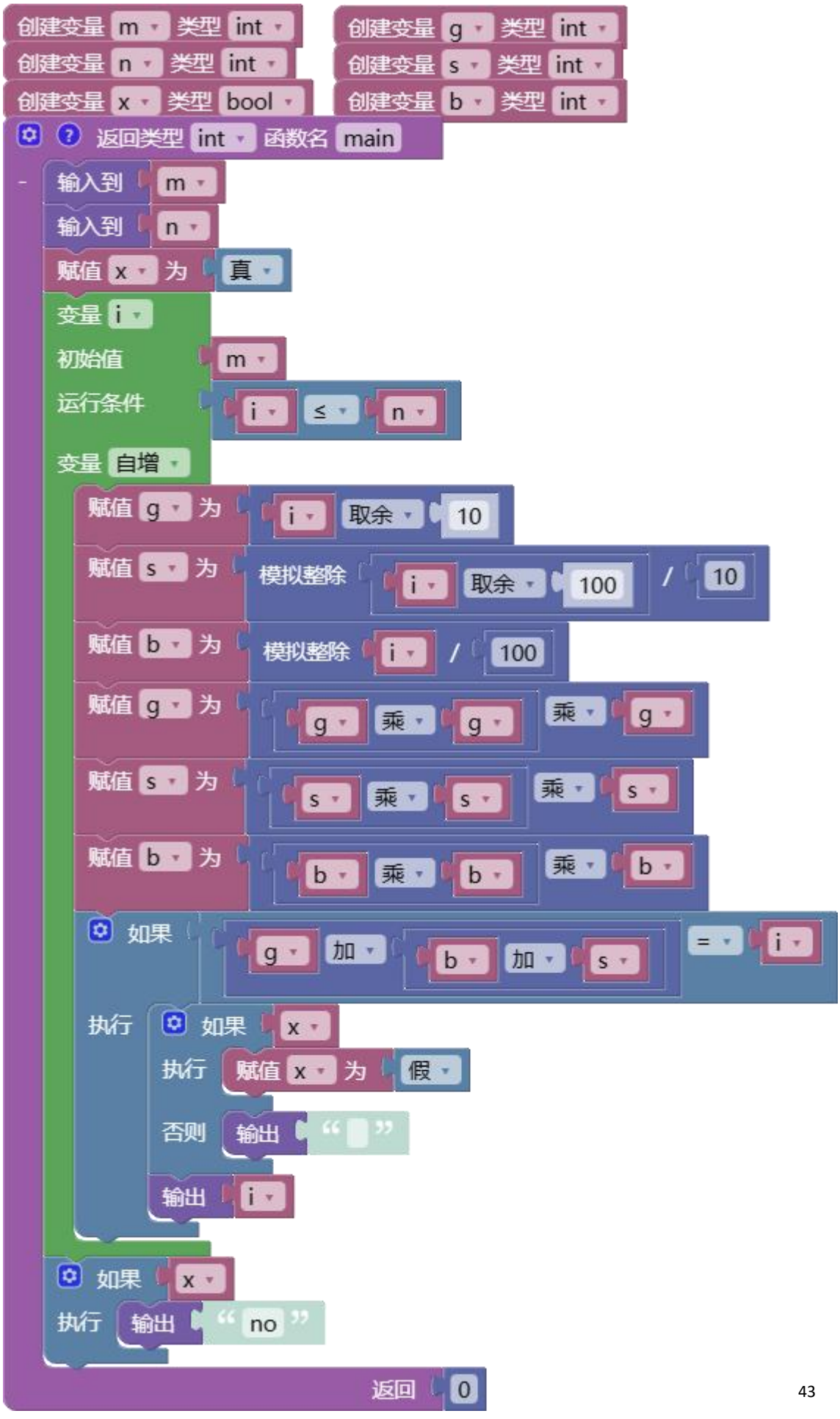
如上图，程序的运行过程如下：

1. 我们遍历取得了 m 到 n 之前的数 i；
2. 创建了三个变量 g、s、b，分别用来保存要数字的三个数位上的数；
3. 计算数字的三次方；
4. 将数字三次方的累加并判断是否和原数相等。

编程技巧小结

- 巧妙的运用了模运算和整除运算取得各位上的数字。

Blockly 题解代码图片


创建变量 m 类型 int
创建变量 n 类型 int
创建变量 x 类型 bool
创建变量 g 类型 int
创建变量 s 类型 int
创建变量 b 类型 int
返回类型 int 函数名 main
输入到 m
输入到 n
赋值 x 为 真
变量 i
初始值 m
运行条件 i ≤ n
变量 自增
赋值 g 为 i 取余 10
赋值 s 为 模拟整除 i 取余 100 / 10
赋值 b 为 模拟整除 i / 100
赋值 g 为 g 乘 g 乘 g
赋值 s 为 s 乘 s 乘 s
赋值 b 为 b 乘 b 乘 b
如果 g 加 b 加 s = i
执行
如果 x
执行 赋值 x 为 假
否则 输出 “ ”
输出 i
如果 x
执行 输出 “ no ”
返回 0

2. 习题练习

完成“C++基础 L1 循环应用（2）”

1.9 循环应用（3）

1. 例题

例题 1. 【质数判定】

题目描述
判定输入的数是不是质数。
输入格式
第一行输入一个整数 t ，1<=t<=1.5*10^4 接下来 t 行，一行一个数 x
输出格式
对于输入的每一个 x，如果 x 是质数输出一行 Y，否则输出一行 N。
样例
输入 #1
```
5
1
2
6
9
100003
```
输出 #1
```
N
Y
N
N
Y
```

习题分析：
我们根据质数的定义：质数只能被 1 和本身整除。整除即两整数相除余数为 0。可以用一个循环列出给从 2 到比该数小 1 之内的所有数，只要有一个能被该数整作，即非质数，全部不能整除即为质数。
进一步优化可以认识到，只需要循环检测 2 到 `sqrt(a)`，即到该数的平方根即可。
代码如下：

C++题解代码

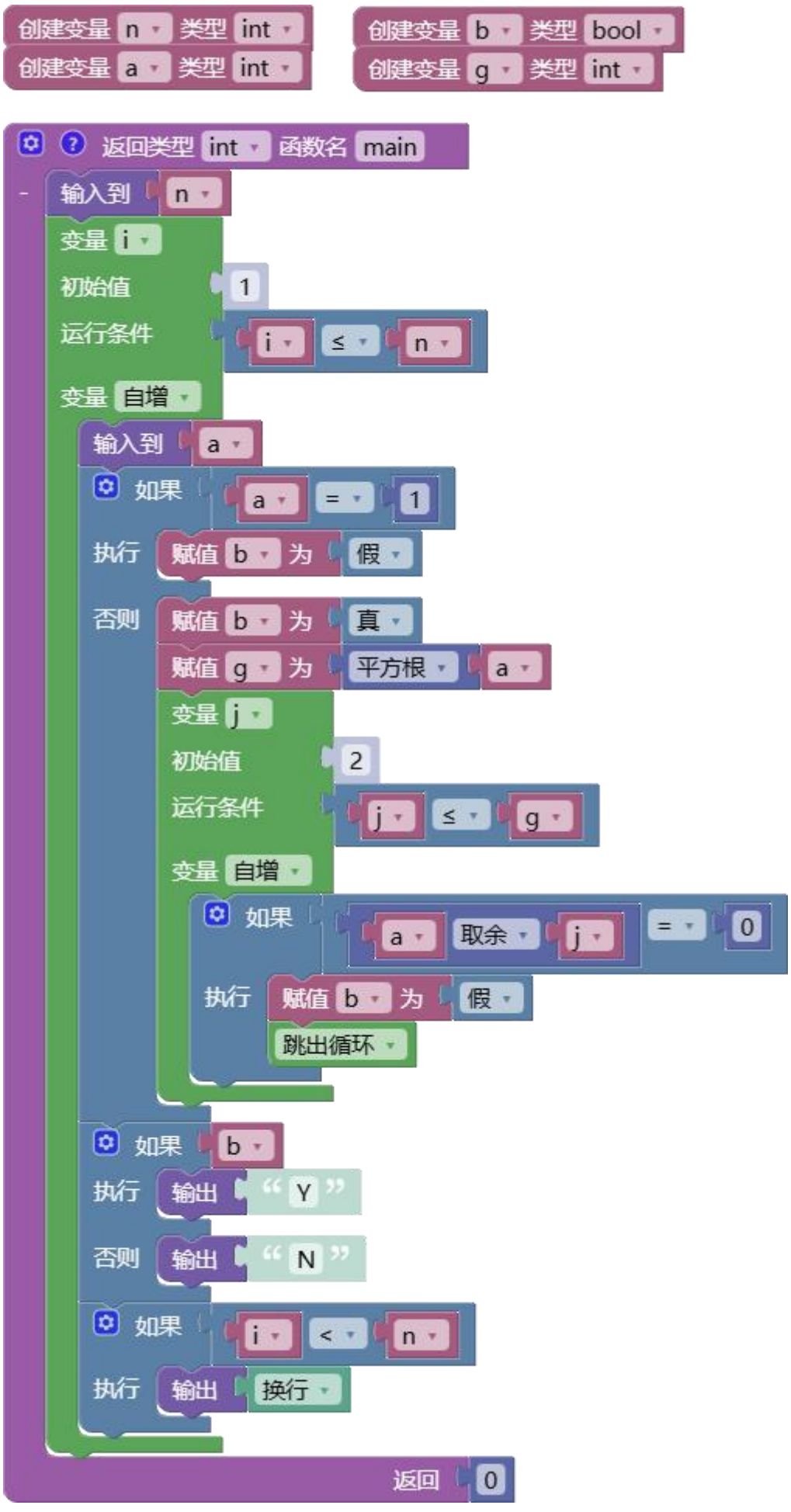
```cpp
#include <bits/stdc++.h>
using namespace std;

int n;
int a;
int g;
```

```
7.  bool b;
8.
9.
10. // The main procedure
11. int main() {
12.   cin>>n;
13.   for (int i = 1; i <= n; i++) {
14.     cin>>a;
15.     if (a == 1) {
16.       b = false;
17.     } else {
18.       b = true;
19.       g = sqrt(a);
20.       for (int j = 2; j <= g; j++) {
21.         if ((a%j) == 0) {
22.           b = false;
23.           break;
24.         }
25.       }
26.     }
27.     if (b) {
28.       cout<<"Y";
29.     } else {
30.       cout<<"N";
31.     }
32.     if (i < n) {
33.       cout<<'\n';
34.     }
35.   }
36.   return 0;
37. }
38.
```

如上图：

- 先输入 n，循环 n 次输入数据到变量 a；
- 考虑到了质数必然大于 1，判断该数是还是等于 1；
- 第二层循环取得从 2 到 `sqrt(a)`之间的数；
- 在循环内判断 a 是否能整除 j，如果能整除，此时用 a 记录遍 x 不是质数，并跳出循环；
- 上面代码中，还有一个变量 b，用来记录最终结果；
- 最终根据 b 分别输出不同内容；

Blockly 题解代码图片


创建变量 n 类型 int
创建变量 a 类型 int
创建变量 b 类型 bool
创建变量 g 类型 int
返回类型 int 函数名 main
输入到 n
变量 i
初始值 1
运行条件 i ≤ n
变量 自增
输入到 a
如果 a = 1
执行 赋值 b 为 假
否则 赋值 b 为 真
赋值 g 为 平方根 a
变量 j
初始值 2
运行条件 j ≤ g
变量 自增
如果 a 取余 j = 0
执行 赋值 b 为 假
跳出循环
如果 b
执行 输出 " Y "
否则 输出 " N "
如果 i < n
执行 输出 换行
返回 0

编程技巧小结

- 根据质数的定义，用代码模拟了质数的判定过程。
- 利用数据知识，优化代码，只循环 a 的平方根次。
- 利用循环枚举产生需要的数据。

2. 习题练习

完成“C++基础 L1 循环应用（3）”