第二讲

Variables, Expression, Graphics

薛浩

2023 年 3 月 16 日

www.stickmind.com

今日话题

话题 1：编程基础 初学编程的新手，一般应该熟练使用函数和库处理字符串相关的编程任务。

话题 2：抽象数据类型的使用 在尝试实现抽象数据类型之前，应该先熟练使用这些工具解决问题。

话题 3：递归和算法分析 递归是一种强有力的思想，一旦掌握就可以解决很多看起来非常难的问题。

话题 4：类和内存管理 使用 C++ 实现数据抽象之前，应先学习 C++ 的内存机制。

话题 5：常见数据结构和算法 在熟练使用抽象数据类型解决常见问题之后，学习如何实现它们是一件很自然的事情。

话题 1：编程基础

初学编程的新手，一般应该熟练使用函数和库处理字符串相关的编程任务。

- C++ 基础
- 函数和库
- 字符串和流

Figure 1: 语音算法

计算机如何存储信息？
屏幕上如何绘制图形？

目录

SimpleCxxLib 介绍

SimpleCxxLib

动机 随书配套库编写的时间较早，除了一些明显的错误，很多现代 C++ 写法没有支持。

升级 创建一个分支 SimpleCxxLib，增加列表初始化、基于范围的 for 循环等，不定期维护。

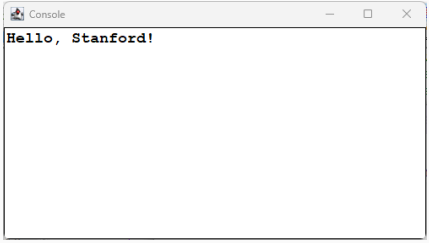
依赖 依赖 Java 运行时环境，前往 Adoptium 下载所需版本并安装。

cppdoc.stickmind.com

Hello With SimpleCxxLib

```
/**
 * File: HelloStanford.cpp
 * -----------------------
 * This file is adapted from the example on page 1 of Kernighan
 * and Ritchie's book The C Programming Language.
 */

#include <iostream>
#include "console.h" // for SimpleCxxLib

int main() {
    std::cout << "Hello, Stanford!" << std::endl;
    return 0;
}
```

Hello With SimpleCxxLib



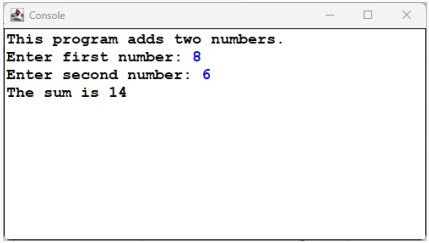
CMake Without SimpleCxxLib

```
cmake_minimum_required(VERSION 3.20)

# Set project information
project(
  HelloStanford
  VERSION 1.0
  DESCRIPTION "Program using
  ↪  SimpleCxxLib"
  LANGUAGES CXX)

# Add an executable target
add_executable(${PROJECT_NAME})
```

```
# Define Cxx standard
set_target_properties(
  ${PROJECT_NAME}
  PROPERTIES CXX_STANDARD 17
             CXX_STANDARD_REQUIRED ON
             CXX_EXTENSIONS OFF)

# Add source files
target_sources(
  ${PROJECT_NAME}
  PRIVATE src/HelloStanford.cpp)
```

CMake With SimpleCxxLib

```
# Fetch SimpleCxxLib
include(FetchContent)
FetchContent_Declare(
  SimpleCxxLib
  # GIT_REPOSITORY https://github.com/xuehao/SimpleCxxLib.git
  GIT_REPOSITORY https://gitee.com/stickmind/SimpleCxxLib.git
  GIT_TAG main)
FetchContent_MakeAvailable(SimpleCxxLib)

# Add libraries
target_link_libraries(${PROJECT_NAME} PRIVATE SimpleCxxLib)

# Copy Java backend
file(COPY ${SimpleCxxLib_SOURCE_DIR}/java/spl.jar
     DESTINATION ${CMAKE_CURRENT_BINARY_DIR})
```


Console
This program adds two numbers.
Enter first number: 8
Enter second number: 6
The sum is 14

Variables 变量

Variables 变量

变量表示计算机上的某个位置，程序通过变量存储信息。

变量包含三个信息：

Variables 变量

变量表示计算机上的某个位置，程序通过变量存储信息。

变量包含三个信息：

Name 用于区分不同的变量

Variables 变量

变量表示计算机上的某个位置，程序通过变量存储信息。

变量包含三个信息：

- Name 用于区分不同的变量
- Type 用于确定变量中存储的信息类型

Variables 变量

变量表示计算机上的某个位置，程序通过变量存储信息。

变量包含三个信息：

Name	用于区分不同的变量
Type	用于确定变量中存储的信息类型
Value	用于确定变量存储的值

Types 类型

类型（Types）包含两个属性：值域和操作集，表示变量可以存储的信息类型。

C++ 中有许多**基本数据类型**（Primitive Types）：

`int`	用于表示正数类型，例如 -1，0，4 等
`double`	用于表示小数，例如 3.14，2.71 等
`bool`	只表示 `true` 和 `false` 两个值
`char`	用于表示单个字符，例如 `'A'`，`'c'`，`'+'` 等

Identifier 标识符

合法的标识符名称需要遵循：

- 名字必须以一个字母或下划线开始
- 名字中所有字符必须是字母、数字或下划线
- 名字不能是 C++ 中的保留字

```
x                                   w
7thBookInTheSeries                  LOUD_AND_PROUD
Harry Potter                        that'sACoolName
noOrdinaryRabbit                    double
lots_of_underscores                 C_19_H_14_O_5_S
```

Declaration 声明变量

C++ 中使用一个变量之前必须先**声明**（Declaration），这样系统才知道该变量的名称、类型、值这些信息。

声明并初始化一个变量的语法：

```
type name = value;
```

例如，

```
int numVotes = 137;
double pricePerPound = 0.93;
```

Input 获取输入

simpio.h

SimpleCxxLib 提供了一些方便编程的接口，利用 simpio.h 接口可以从用户获取输入，创建交互式应用。

例如，

```
#include "simpio.h"

int numVotes = getInteger("How many votes? ");
double pricePerPound = getReal("What is the price? ");
```

注意，字符串的末尾为什么多了一个空格?

Expressions 表达式

Expressions 表达式

表达式由操作数（operand）和运算符（operator）组成，表示一个值。

常见的算术运算符有：

- `+` 加法
- `-` 减法
- `*` 乘法
- `/` 除法

Precedence 优先级

下述写法并不能求出两个数的平均值：

```
a + b / 2
```

算术运算符遵循以下优先级：

运算符	优先级
( )	高
* / %	
+ -	低

同级别的算术运算符遵循从左向右的结合性。

Precedence 优先级

下述写法并不能求出两个数的平均值：

```
(a + b) / 2
```

算术运算符遵循以下优先级：

运算符	优先级
()	高
* / %	
+ -	低

同级别的算术运算符遵循从左向右的结合性。

Mod 模运算

特殊运算符 % 称为模运算符（mod operator），用于计算一个值除以另一个值的余数。

a % b 读作“a 对 b 取模”

例如，

```
15 % 3 -> 0
14 % 8 -> 6
21 % 2 -> 1
```

整数除法

C++ 中两个整数相除，结果将会向下取整（rounding down）。

例如，下述代码输出将是 3:

```
int a = 3;
int b = 4;
int average = (a + b) / 2;
std::cout << average << std::endl;
```

浮点数除法

C++ 中两个浮点数相除，结果将是正确的。

表达式中，任何一个操作数是浮点类型，结果都将是正确的。

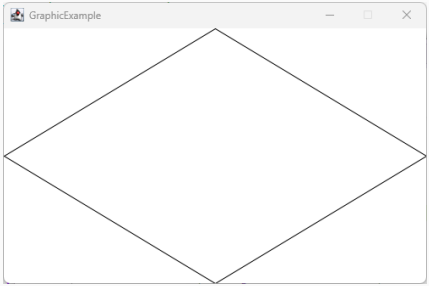
对于两个整型的平均值，可以这样写：

```
int a = 3;
int b = 4;
double average = (a + b) / 2.0;
std::cout << average << std::endl;
```

AddTwoNumbers.cpp

```cpp
/*
 * File: AddTwoNumbers.cpp
 * -----------------------
 * This program adds two floating-point
↪  numbers and prints their sum.
 */

#include <iostream>
#include "console.h"  // for console
#include "simpio.h"   // for getInteger
using namespace std;

int main() {
  cout << "This program adds two
↪  numbers." << endl;
  // Read two values from the user.
  int n1 = getInteger("Enter first
↪  number: ");
  int n2 = getInteger("Enter second
↪  number: ");

  // Compute their sum.
  int sum = n1 + n2;

  // Print out the summation.
  cout << "The sum is " << sum << endl;

  return 0;
}
```

Graphics 绘图


GraphicExample

gwindow.h

C++ 是一门面向对象语言，除了基本数据类型外，还有大量自定义数据类型。自定义数据类型称为**类**（class），声明的变量称为**对象**（object）。

利用 SimpleCxxLib 可以创建 GWindow 对象进行图形绘制相关的操作。

例如，创建一个 GWindow 对象：

```
#include "gwindow.h"
GWindow window;
```

Method 方法调用

应用于对象的操作称为**方法**（method）。在对象上调用一个方法，可以使用语法:

```
object.method(parameters)
```

例如，作为**表达式**的方法调用

```
double width = window.getWidth();
double height = window.getHeight();
```

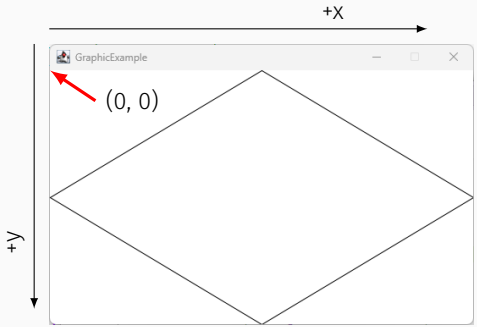
例如，作为**消息**的方法调用

```
window.setWindowTitle("Diamond");
window.drawLine(0, 0, 30, 30);
```

图形坐标

GWindow 对象通过定义 *x* 和 *y* 坐标来定位。

- *x* 从左向右增长；*y* 从上向下增长
- *x* 和 *y* 都以像素为单位



Diamond.cpp

```
/**
 * File: GraphicExample.cpp
 * ------------------------
 * This program illustrates the use of
↪  graphics using the GWindow class.
 */

#include "gwindow.h"

int main() {
  GWindow window;

  /* Set the window title */
  window.setWindowTitle("Diamond");

  /* Get the width and height of  */
  double width = window.getWidth();
  double height = window.getHeight();

  /* Draw line connecting the midpoints
↪  of the edges. */
  window.drawLine(0, height / 2, width /
↪  2, 0);
  window.drawLine(width / 2, 0, width,
↪  height / 2);
  window.drawLine(width, height / 2,
↪  width / 2, height);
  window.drawLine(width / 2, height, 0,
↪  height / 2);

  return 0;
}
```

gobjects.h

通过**图形对象**（Graphics Objects）可以更方便的创建图形应用。

创建一个图形对象分两个步骤：

- 声明一个指针类型的变量来记录对象的位置
- 使用关键字 new 来创建对象

指针变量常用于引用较大的数据结构以节省空间，声明指针变量需要做的就是在类型后面加个星号（*）。

例如，

```
#include "gobjects.h"
GLine* line = new GLine(0, height / 2, width / 2, 0);
```

Assignment 赋值运算符

变量包含三个信息：名称、类型、值。一旦声明变量，名称和类型将无法改变。但是，我们可以修改变量的值。

修改一个变量值的语句：

```
name = newValue;
```

该语句称为**赋值语句**（Assignment Statement），这里的 = 称为赋值运算符。

计算机如何存储信息？
屏幕上如何绘制图形？

总结


Operand
Object
Method
getInteger
Class
Expression
Type
Operator
GWindow
Identifier
Down
Rounding
Variable
Mod
getReal
Graphics

问题?