基于 Linux 的 C++

第三讲　函 数

提　纲

函数声明与调用

函数定义

函数调用规范

函数声明与调用

函数调用

主调（客户）函数与被调（服务器）函数

函数调用时的参数与返回值

例一：Swap(a, b);

例二：n = Add(a, b);

函数声明与调用

函数原型

函数的实现与调用格式说明：作为函数接口

一般出现在头文件中

格式：函数返回值类型 函数名称(形式参数列表);

例一：int Add(int x, int y);

例二：void Swap(int x, int y);

例三：void Compute();

函数定义

函数实现

函数定义，使用编程语言给出函数的执行步骤

函数返回值

函数完成后带回来的结果

主调函数可以使用

谓词函数

返回 bool 类型值的函数

表达某项任务是否完成或某个条件是否满足

函数重载

Add 函数

编写函数 Add，求两个整数之和

```
int Add( int x, int y )
{
  int t;
  t = x + y;
  return t;
}
```

Compare 函数

编写函数 Compare，比较两个整型数据 x、y 的大小。若 x 等于 y 返回 0，若 x 大于 y 返回 1，若 x 小于 y 返回 -1

```
int Compare( int x, int y )
{
  int t;
  if( x == y )
    t = 0;
  else if( x > y )
    t = 1;
  else
    t = -1;
  return t;
}
```

Swap 函数

编写函数 Swap，互换两个整型数据 x、y 的值

```
void Swap( int x, int y )
{
  int t;
  t = x;
  x = y;
  y = t;
  return;    //  因函数没有返回值，只需直接列写 return 语句
}
```

Swap 函数

挑战问题：如何在不使用临时中转变量的情况下互换两个整型变量的值？

多条 return 语句

编写函数 Compare，比较两个整型数据 x、y 的大小。若 x 等于 y 返回 0，若 x 大于 y 返回 1，若 x 小于 y 返回 -1

```
// 允许函数中包含多条 return 语句
// 函数在执行到第一条 return 语句后终止
int Compare( int x, int y )
{
  if( x == y )
    return 0;
  else if( x > y )
    return 1;
  else
    return -1;
}
```

谓词函数

编写函数 IsLeap，判断某个给定年份是否为闰年

```
bool IsLeap( int year )
{
  return year % 4 == 0 && year % 100 != 0 || year % 400 == 0;
}
```

函数重载

定义同名但参数不完全相同的函数

示　例

```
int Max( int x, int y );
char Max( char x, char y );
bool Max( bool x, bool y );
```

函数调用规范

函数调用示例

参数传递机制：值传递与引用传递

函数调用栈框架

函数调用示例

编写程序将用户输入的两个整数相加，要求尽可能使用函数将程序中的操作独立出来

```
#include <iostream>
using namespace std;
void Welcome();
int GetInteger( int idx );
int Add( int x, int y );
int main(){
  int a, b, sum;
  Welcome();
  a = GetInteger( 1 );
  b = GetInteger( 2 );
  sum = Add( a, b );
  cout << "The sum is" << sum << "." << endl;
  return 0;
}
```

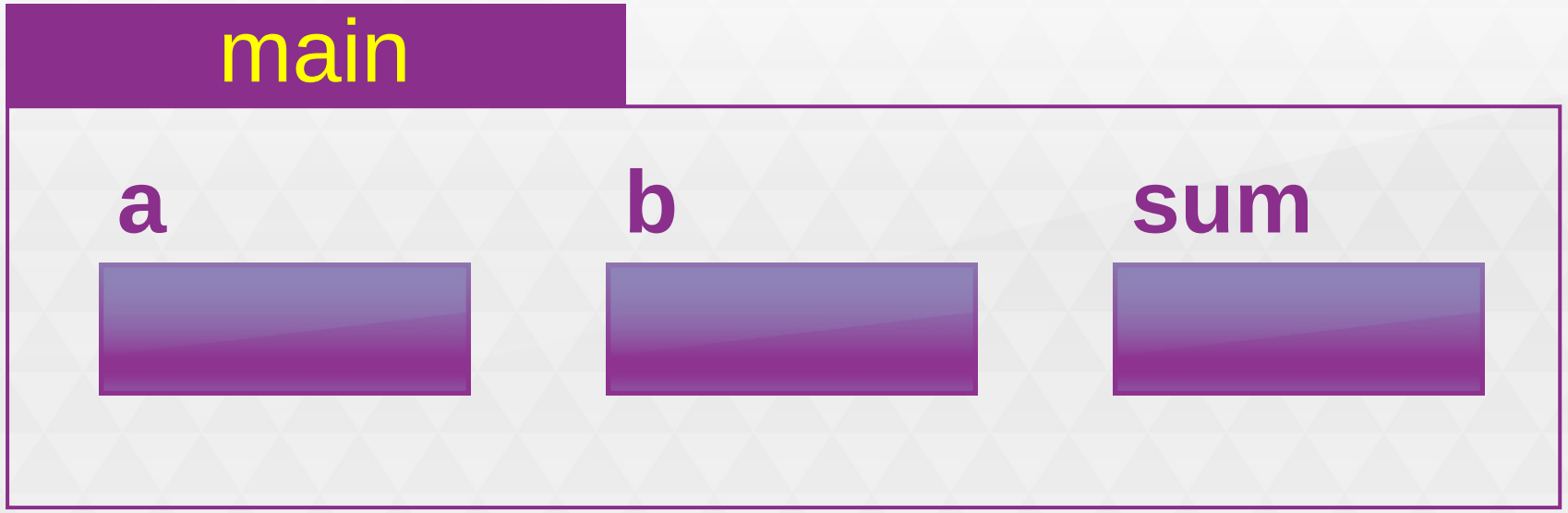
函数调用示例

```
void Welcome()
{
  cout << "The program gets two integers, and prints their sum." << endl;
}
int GetInteger( int idx )
{
  int t;
  cout << "No. " << idx << ": ";
  cin >> t;
  return t;
}
int Add( int x, int y )
{
  int t;
  t = x + y;
  return t;
}
```

值传递机制

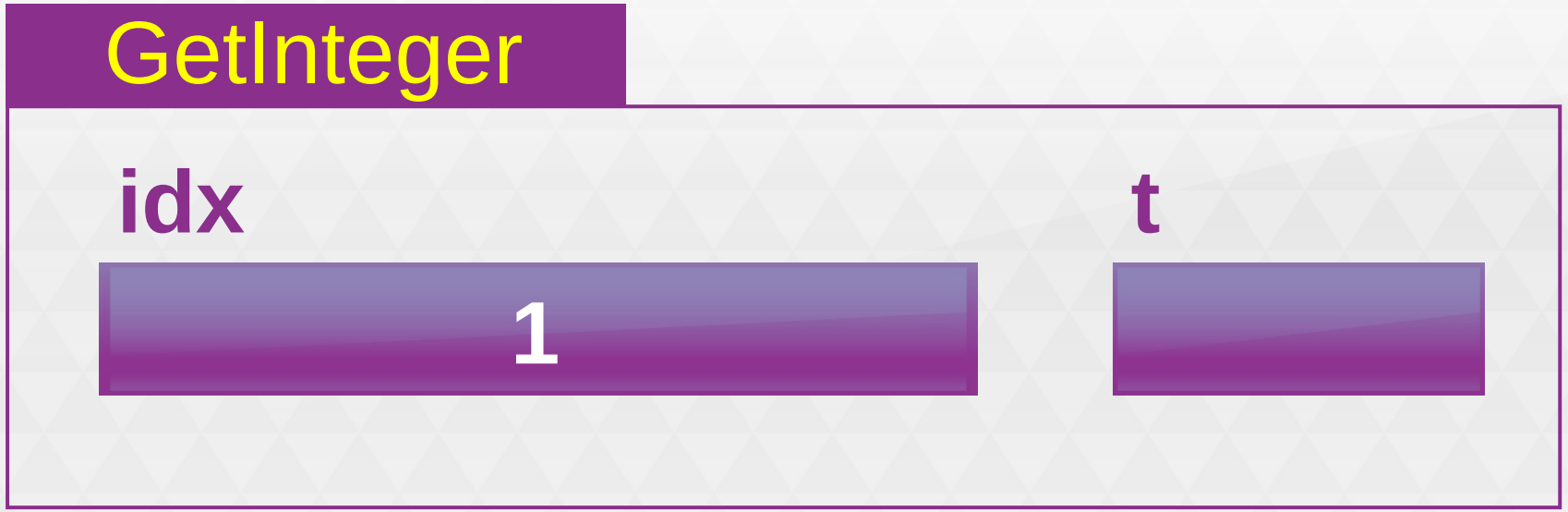
- 形式参数在函数调用时才分配存储空间，并接受实际参数的值
- 实际参数可以为复杂的表达式，在函数调用前获得计算
- 形式参数与实际参数可同名，也可不同名
- 参数较多时，实际参数值逐一赋值，它们必须保持数目、类型、顺序的一致
- 值的复制过程是单向不可逆的，函数内部对形式参数值的修改不会反映到实际参数中去
- 函数参数一般为函数输入集的一部分，函数输出集一般使用返回值表示，只有使用特殊的手段才可以将函数参数作为函数输出集的一部分

函数调用栈框架



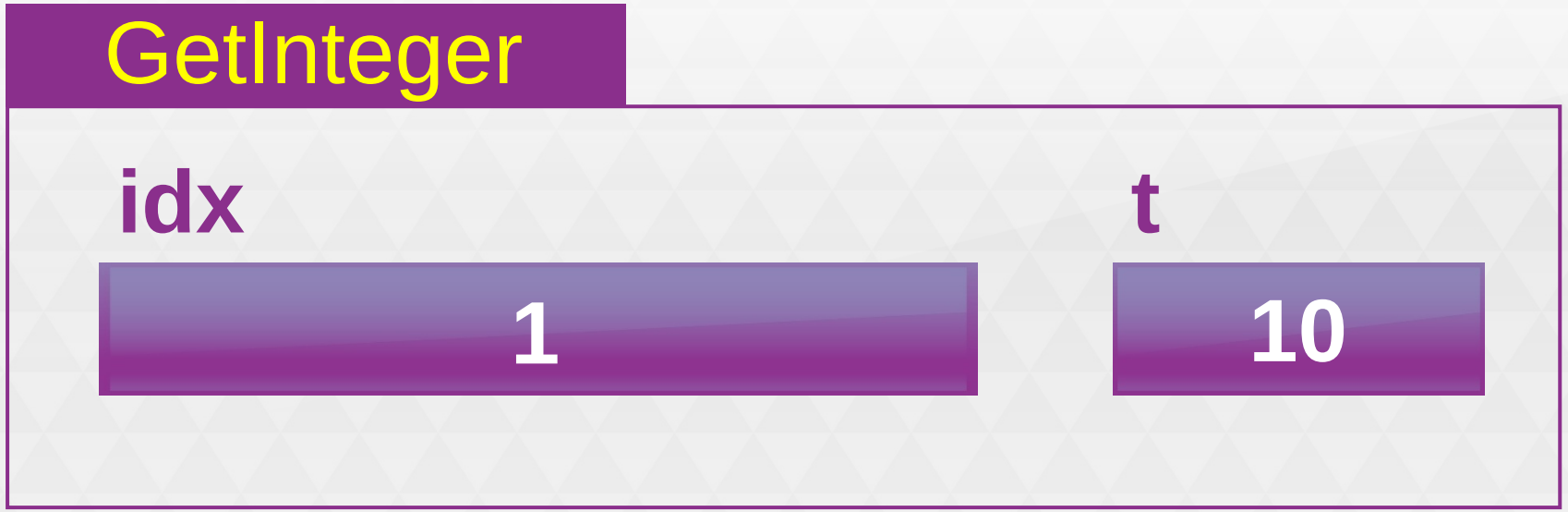
首次调用 GetInteger 函数前

函数调用栈框架



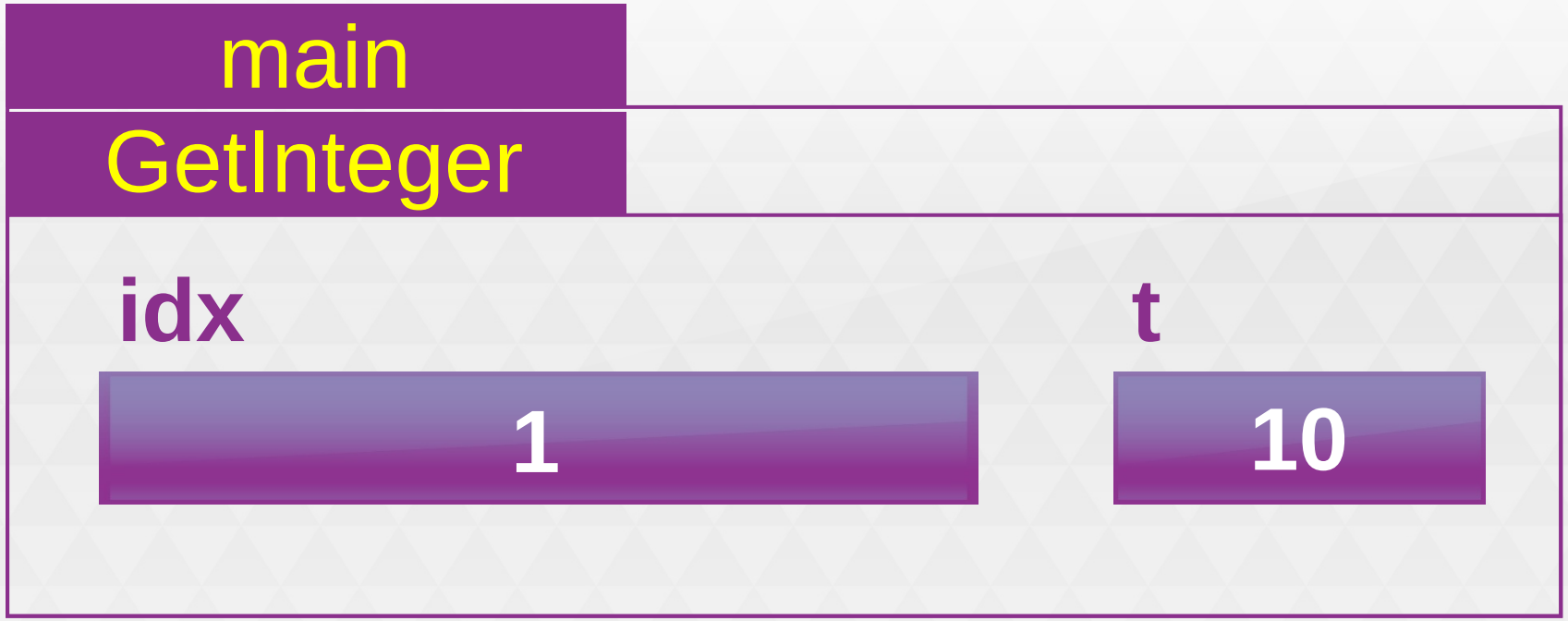
首次调用 GetInteger 函数，数据输入前

函数调用栈框架



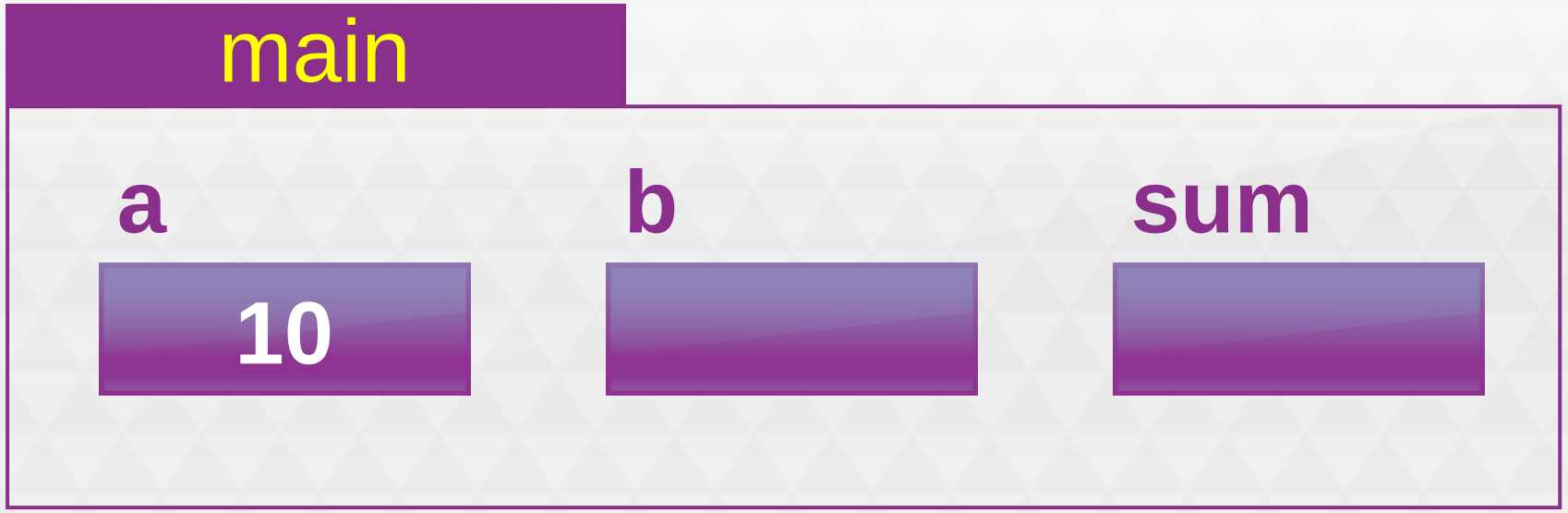
首次调用 GetInteger 函数，数据输入后

函数调用栈框架



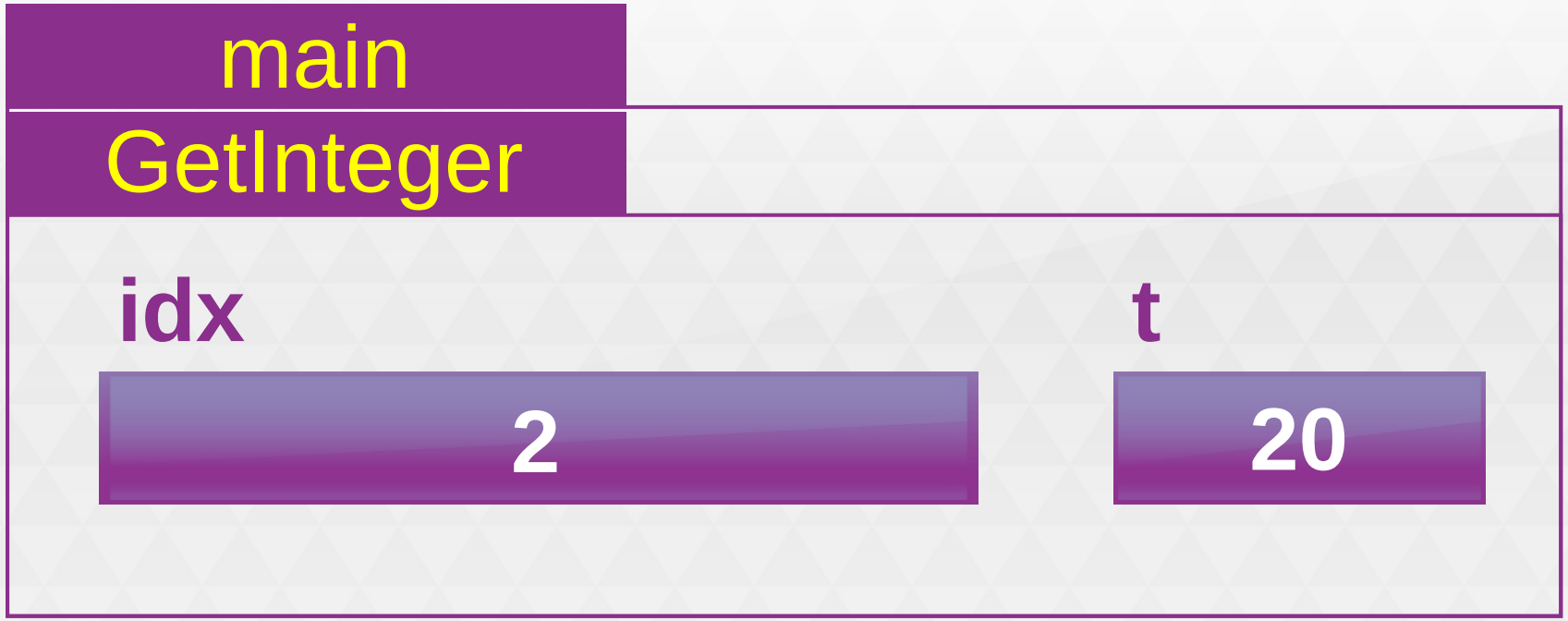
首次调用 GetInteger 函数，数据输入后

函数调用栈框架



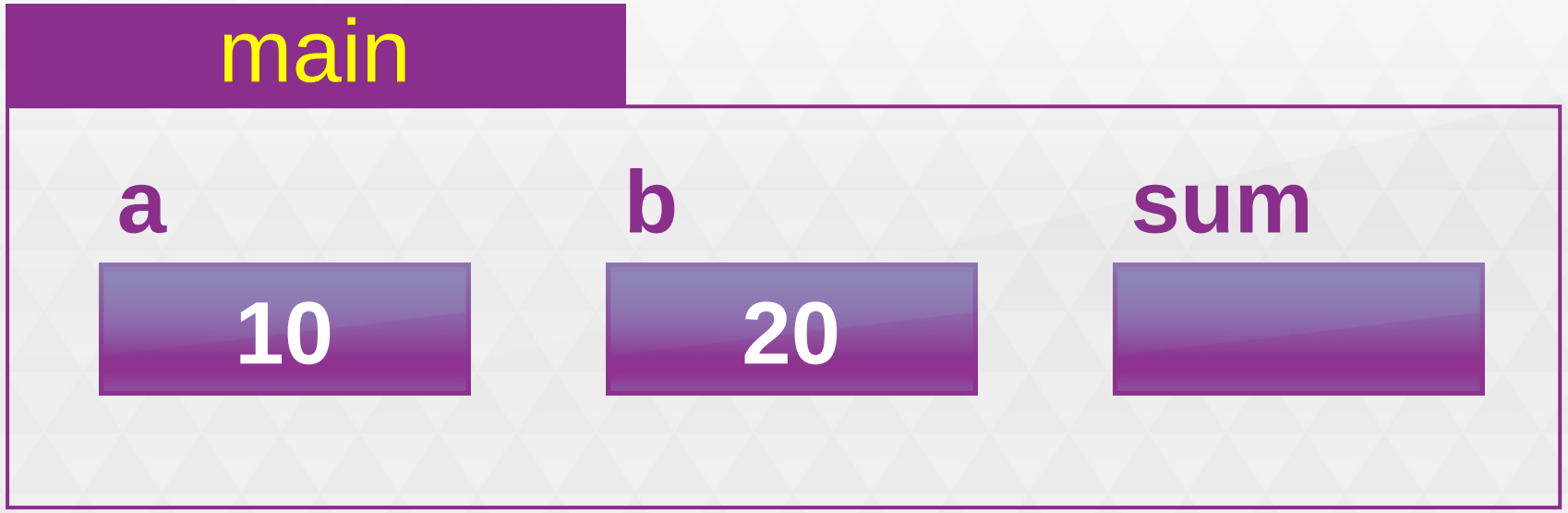
首次调用 GetInteger 函数结束

函数调用栈框架



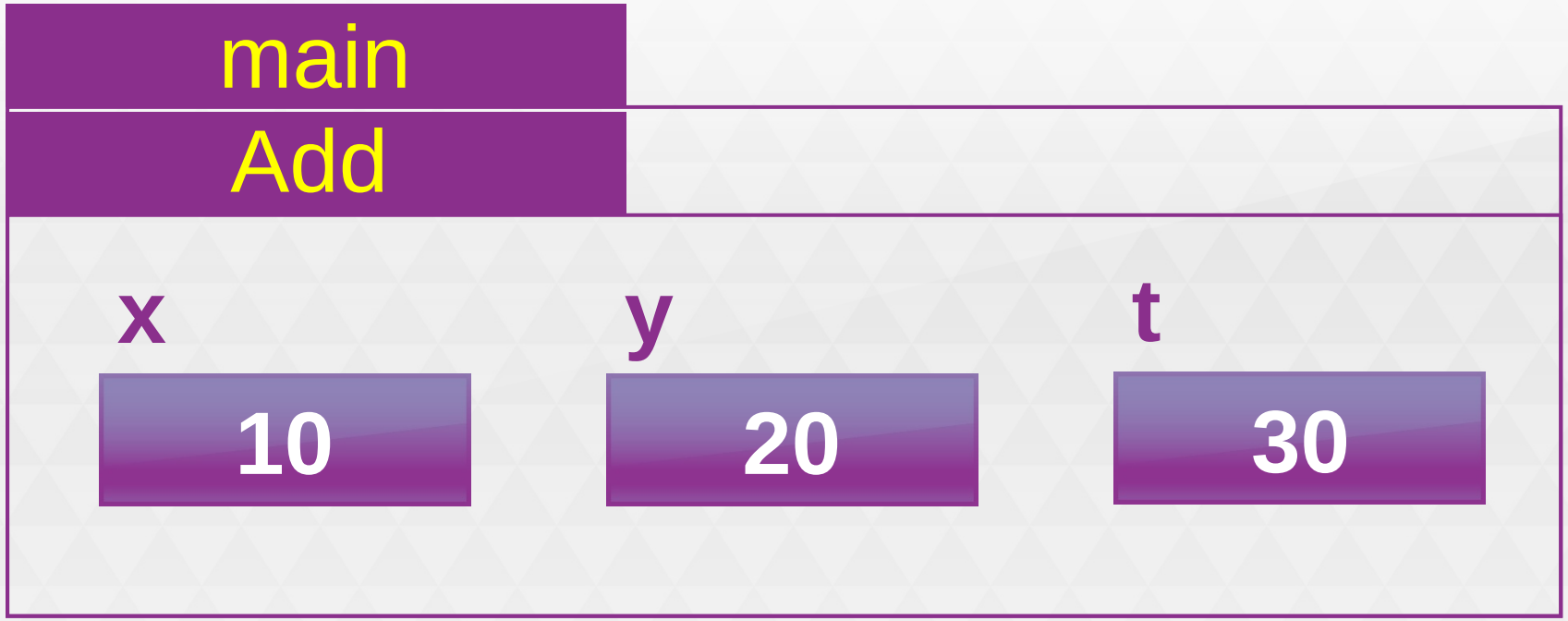
再次调用 GetInteger 函数，数据输入后

函数调用栈框架



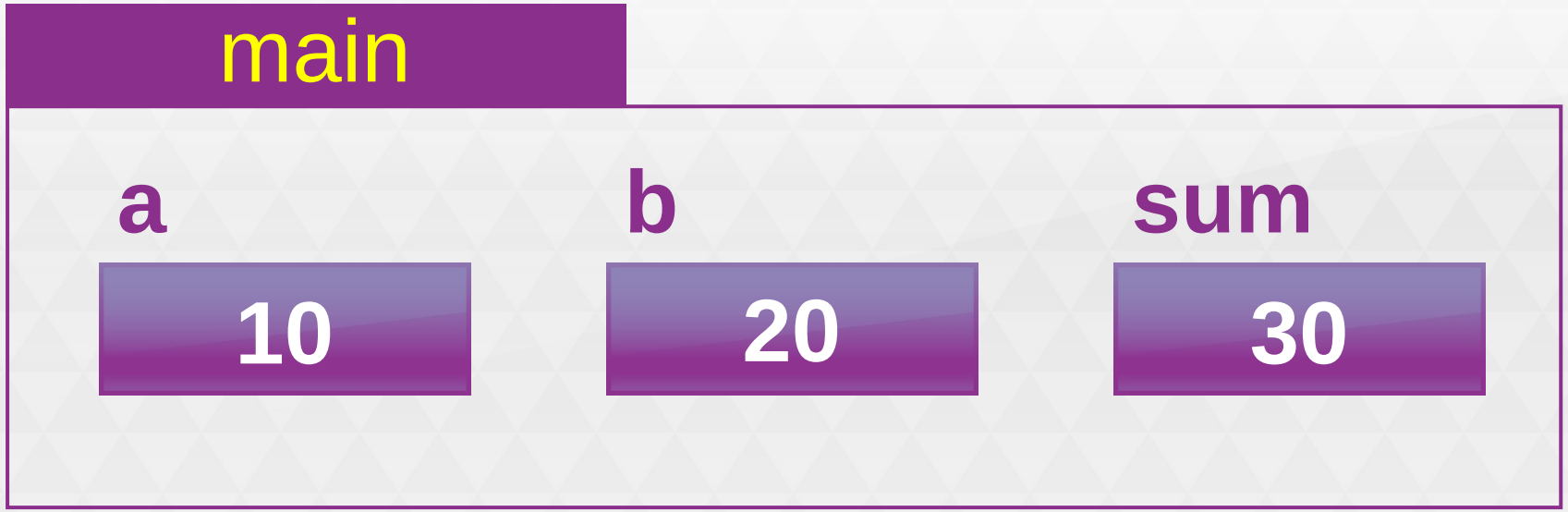
再次调用 GetInteger 函数结束

函数调用栈框架



调用 Add 函数

函数调用栈框架



调用 Add 函数结束，结果为 30

整数互换示例第一版

编写程序将用户输入的两个整数互换

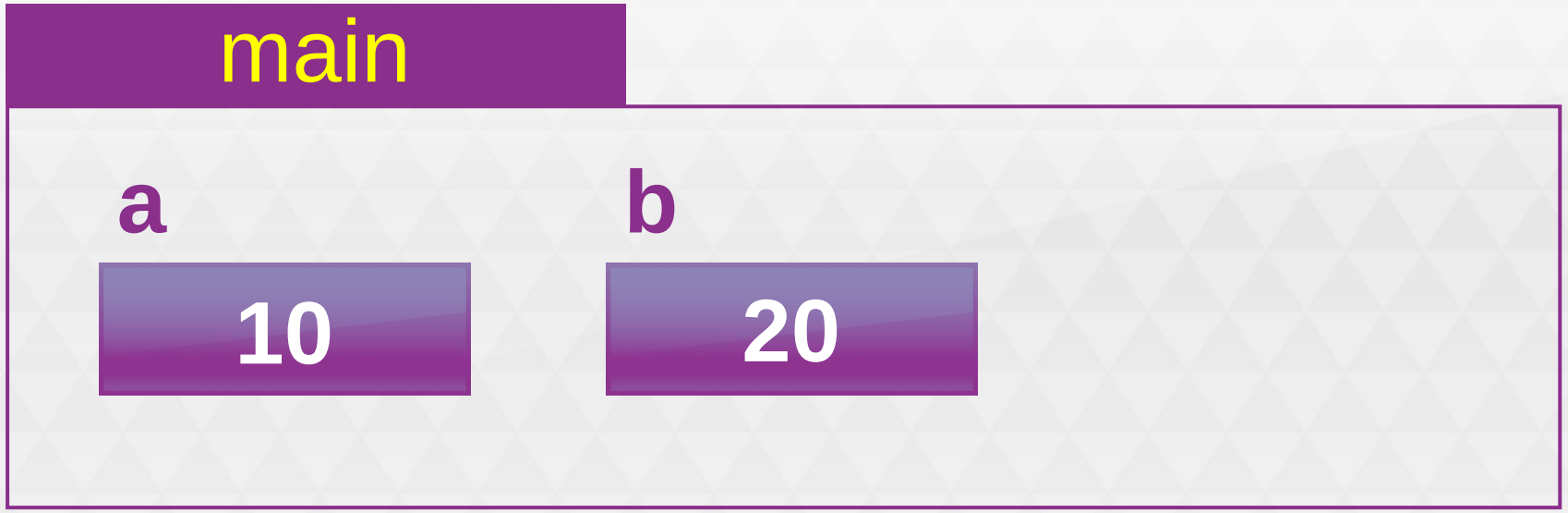
```
#include <iostream>
using namespace std;
void Welcome();
int GetInteger( int idx );
void Swap( int x, int y );

int main()
{
  int a, b;
  /* 输入部分 */
  Welcome();
  a = GetInteger( 1 );
  b = GetInteger( 2 );
  /* 数据处理与输出部分 */
  cout << "In main():  a: " << a << "; b: " << b << endl;
  Swap(a, b);
  cout << "In main():  a: " << a << "; b: " << b << endl;
  return 0;
}
```

整数互换示例第一版

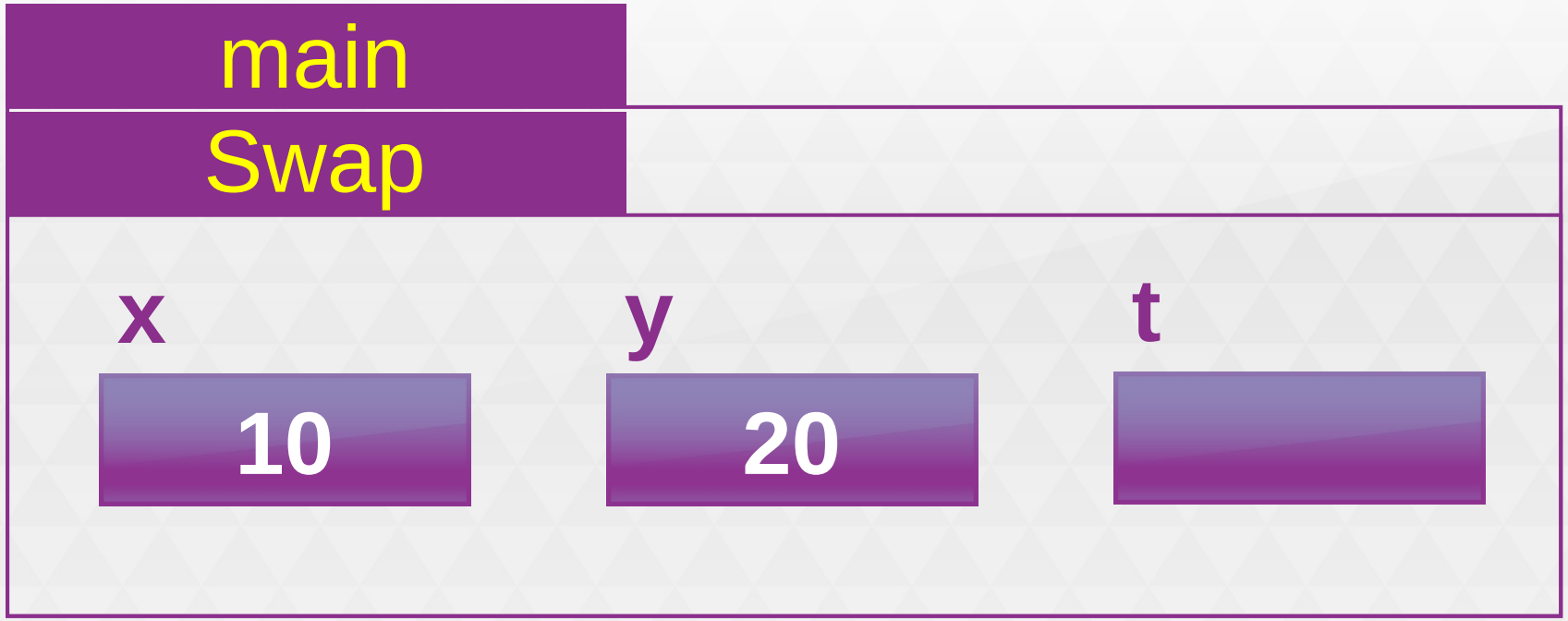
```
void Welcome()
{
  cout << "The program gets two integers, and tries to swap them.\n";
}
int GetInteger( int idx )
{
  int t;
  cout << "No. " << idx << ": ";
  cin >> t;
  return t;
}
void Swap( int x, int y )
{
  int t;
  cout << "In Swap():  x: " << x << "; y: " << y << endl;
  t = x;
  x = y;
  y = t;
  cout << "In Swap():  x: " << x << "; y: " << y << endl;
  return;
}
```

整数互换程序栈框架



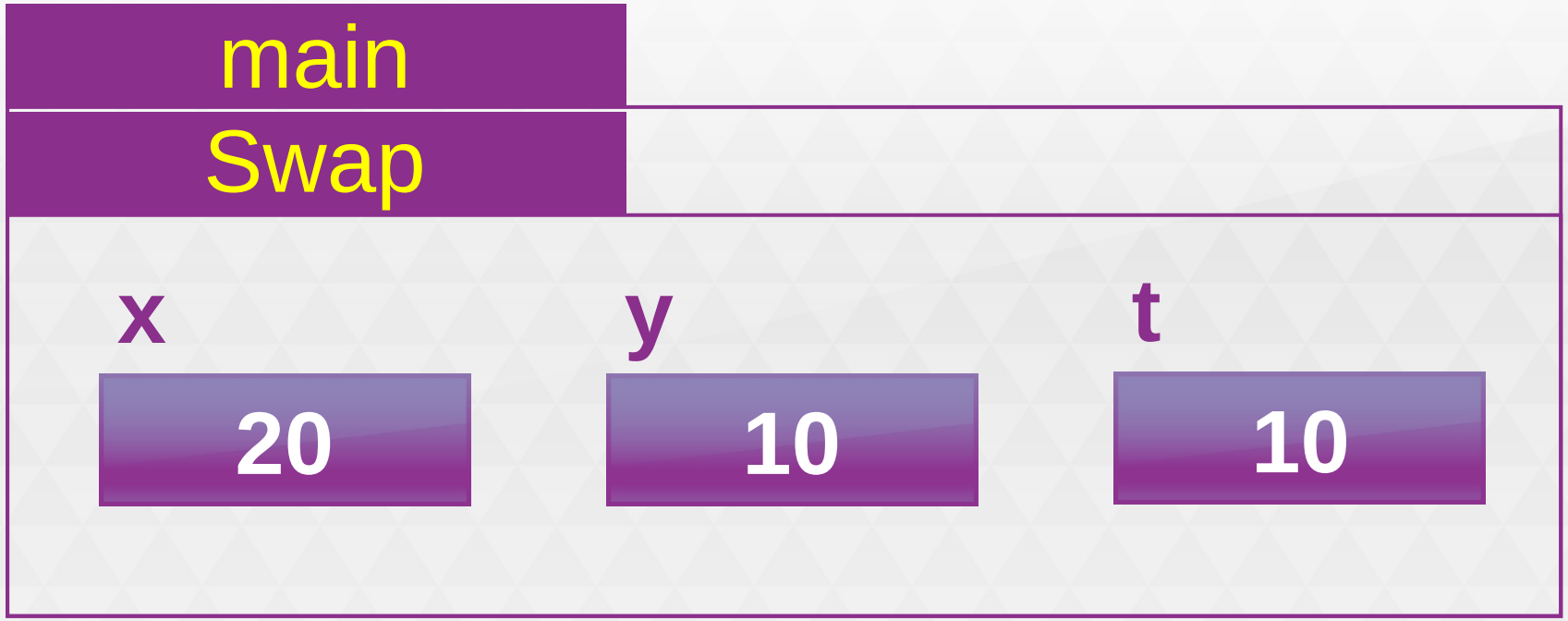
调用 Swap 函数前

整数互换程序栈框架



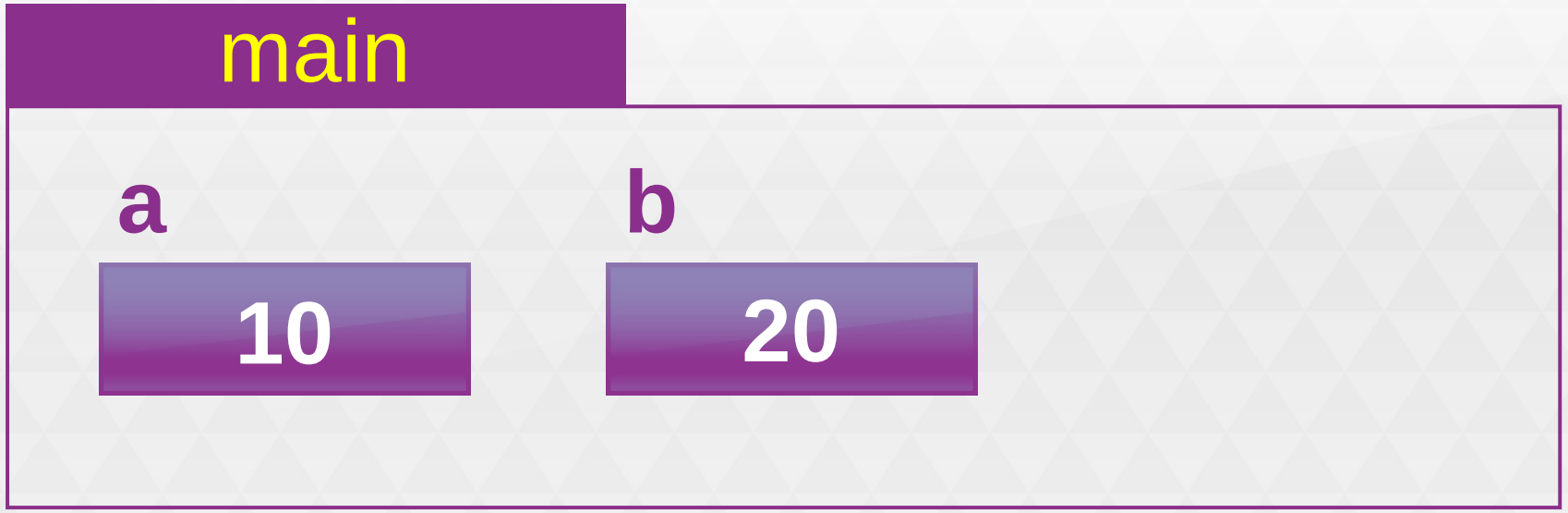
调用 Swap 函数，数据互换前

整数互换程序栈框架



调用 Swap 函数，数据互换后

整数互换程序栈框架



调用 Swap 函数后

整数互换示例第二版

编写程序将用户输入的两个整数互换

```
#include <iostream>
using namespace std;
int a, b;    /* 全局数据对象声明，以保证所有函数都可以访问这两个数据对象 */
void Welcome();
int GetInteger( int idx );
void Swap();    /* 不再需要函数参数  */

int main()
{
  /* 输入部分 */
  Welcome();
  a = GetInteger( 1 );
  b = GetInteger( 2 );
  /* 数据处理与输出部分 */
  cout << "In main():  a: " << a << "; b: " << b << endl;
  Swap();
  cout << "In main():  a: " << a << "; b: " << b << endl;
  return 0;
}
```

整数互换示例第二版

```
void Welcome()
{
  cout << "The program gets two integers, and tries to swap them.\n";
}
int GetInteger( int idx )
{
  int t;
  cout << "No. " << idx << ": ";
  cin >> t;
  return t;
}
void Swap()
{
  int t;
  cout << "In Swap():  x: " << x << "; y: " << y << endl;
  t = a;
  a = b;
  b = t;
  cout << "In Swap():  x: " << x << "; y: " << y << endl;
  return;
}
```

编程实践

3.1 编写函数IsPrime，判断某个大于2的正整数是否为素数。

3.2 编写函数gcd与lcm，分别求两个正整数的最大公约数与最小公倍数。