PROGRAMMING WITH C++

LAB 6

Xi’an Jiaotong-Liverpool University
西交利物浦大学

Example 1: Displaying Prime Numbers

Last lab, we write **program** that displays the first 50 prime numbers in five lines, each of which contains 10 numbers. An integer greater than 1 is *prime* if its only positive divisor is 1 or itself. For example, 2, 3, 5, and 7 are prime numbers, but 4, 6, 8, and 9 are not.

Solution: The problem can be broken into the following tasks:

- For number = 2, 3, 4, 5, 6, ..., test whether the number is prime.
- Determine whether a given number is prime.
- Count the prime numbers.
- Print each prime number, and print 10 numbers per line.

Problem: write **Function** that displays the first 50 prime numbers in five lines, each of which contains 10 numbers.

Example 2: Maximum Number Overloading Functions

- Previously, the max function works only with the int data type.
- But what if you need to find which of two floating-point numbers has the maximum value?
- The solution is to create another function with the same name but different parameters

Example 3: Pass by Value

When you invoke a function with a parameter, the value of the argument is passed to the parameter.

This is referred to as *pass-by-value*. If the argument is a variable rather than a literal value, the value of the variable is passed to the parameter.

The variable is not affected, regardless of the changes made to the parameter inside the function.

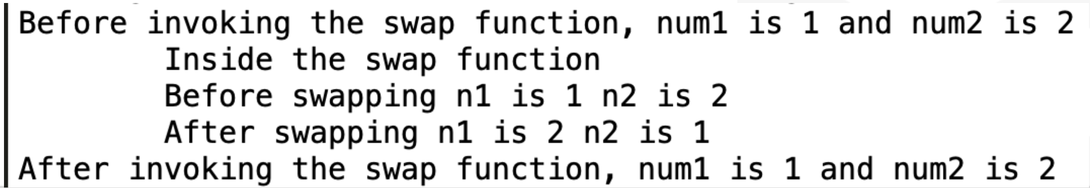
Pass-by-value has serious limitations. Here gives a program that shows the effect and limitation of passing by value. The program creates a function for swapping two variables. The swap function is invoked by passing two arguments. Interestingly, the values of the arguments are not changed after the function is invoked.

Example 3: Pass by Value

SwapByValue.cpp

```
#include <iostream>
using namespace std;
// Attempt to swap two variables does
not work!
void swap(int n1, int n2)
{
  cout << "\tInside the swap function" <<
endl;
  cout << "\tBefore swapping n1 is " << n1
<<
    " n2 is " << n2 << endl;

  // Swap n1 with n2
  int temp = n1;
  n1 = n2;
  n2 = temp;
  cout << "\tAfter swapping n1 is " << n1
<<
    " n2 is " << n2 << endl;
}
```

```
int main()
{
  // Declare and initialize variables
  int num1 = 1;
  int num2 = 2;

  cout << "Before invoking the swap function,
num1 is "
    << num1 << " and num2 is " << num2 <<
endl;
  // Invoke the swap function to attempt to
swap two variables
  swap(num1, num2);

  cout << "After invoking the swap function,
num1 is " << num1 <<
    " and num2 is " << num2 << endl;

  return 0;
}
```



```
Before invoking the swap function, num1 is 1 and num2 is 2
        Inside the swap function
        Before swapping n1 is 1 n2 is 2
        After swapping n1 is 2 n2 is 1
After invoking the swap function, num1 is 1 and num2 is 2
```

Example 4: Passing Arrays to Functions

Just as you can pass single values to a function, you can also pass an entire array to a function. This gives an example to demonstrate how to declare and invoke this type of functions.

```
#include <iostream>
using namespace std;

void printArray(int list[], int arraySize); // Function prototype

int main()
{
  int numbers[6] = {1, 4, 3, 6, 8, 9};
  printArray(numbers, 6); // Invoke the function

  return 0;
}

void printArray(int list[], int arraySize)
{
  for (int i = 0; i < arraySize; i++)
  {
    cout << list[i] <<  " "<<endl;
  }
}
```

```
1
4
3
6
8
9
```

Example 5: Print Calendar Case Study

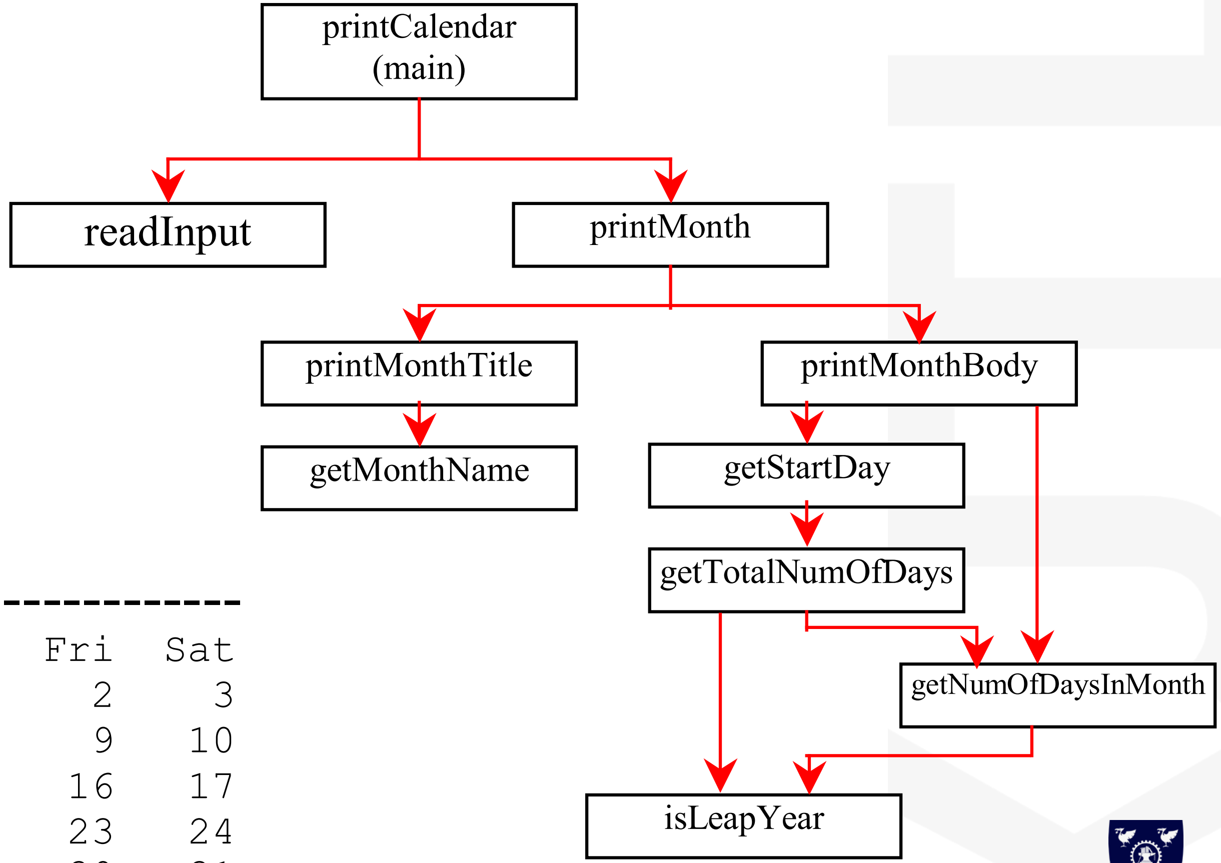
STEPWISE REFINEMENT

The concept of Function abstraction can be applied to the process of developing programs. When writing a large program, you can use the “divide and conquer” strategy, also known as *stepwise refinement*, to decompose it into subproblems.

The subproblems can be further decomposed into smaller, more manageable problems.

Example 5: Print Calendar Case Study

Design the PrintCalendar to demonstrate the stepwise refinement approach.



printCalendar (main)

readInput

printMonth

printMonthTitle

printMonthBody

getMonthName

getStartDay

getTotalNumOfDays

getNumOfDaysInMonth

isLeapYear

```
            December 2005
--------------------------------------
Sun   Mon   Tue   Wed   Thu   Fri   Sat
                          1     2     3
  4     5     6     7     8     9    10
 11    12    13    14    15    16    17
 18    19    20    21    22    23    24
 25    26    27    28    29    30    31
```

PrintCalendar.cpp

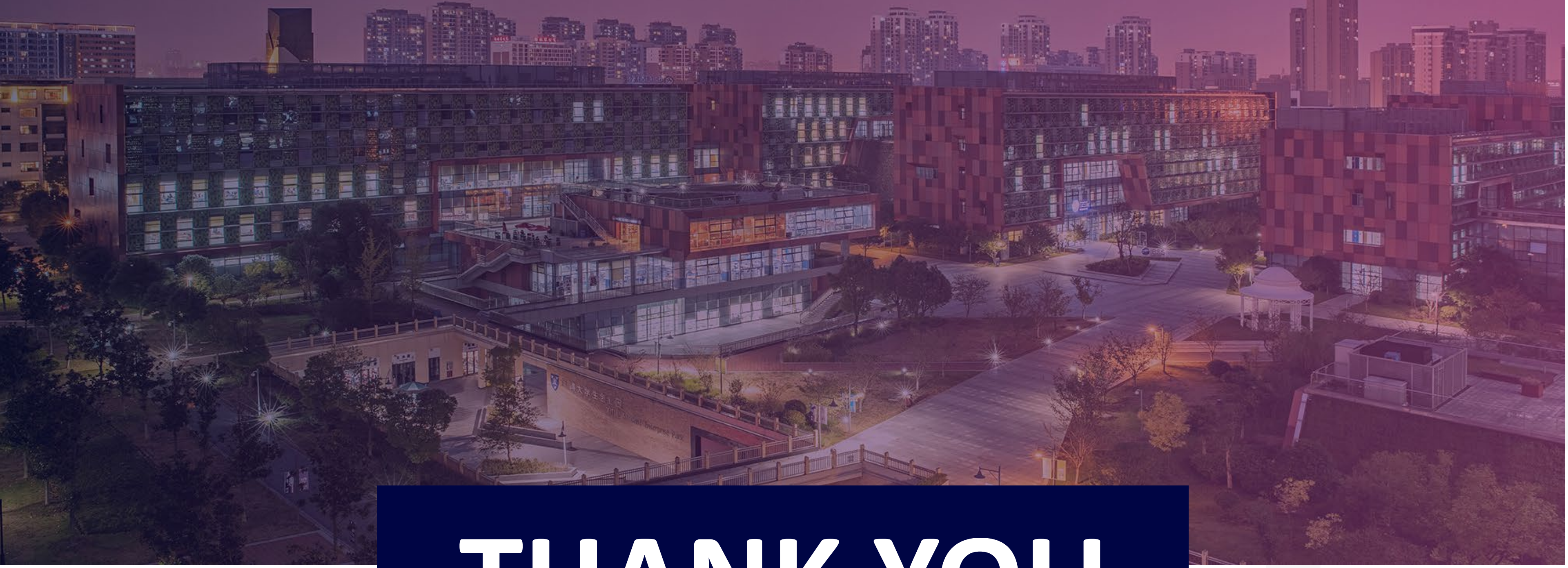
```
#include <iostream>
#include <iomanip>
using namespace std;
// Function prototypes
void printMonth(int year, int month);
void printMonthTitle(int year, int month);
void printMonthName(int month);
void printMonthBody(int year, int month);
int getStartDay(int year, int month);
int getTotalNumberOfDays(int year, int month);
int getNumberOfDaysInMonth(int year, int month);
bool isLeapYear(int year);
int main(){
  // Prompt the user to enter year
  cout << "Enter full year (e.g., 2001): ";
  int year;
  cin >> year;
  // Prompt the user to enter month
  cout << "Enter month in number between 1 and 12: ";
  int month;
  cin >> month;
  // Print calendar for the month of the year
  printMonth(year, month);
  return 0;}
```

```
// Print the calendar for a month in a year
void printMonth(int year, int month)
{
  // Print the headings of the calendar
  printMonthTitle(year, month);

  // Print the body of the calendar
  printMonthBody(year, month);
}

// Print the month title, e.g., May, 1999
void printMonthTitle(int year, int month)
{
  printMonthName(month);
  cout << " " << year << endl;
  cout << "-----------------------------" << endl;
  cout << " Sun Mon Tue Wed Thu Fri Sat" << endl;
}

// Get the number of days in a month
int getNumberOfDaysInMonth(int year, int month)
{
  if (month == 1 || month == 3 || month == 5 || month == 7 ||
    month == 8 || month == 10 || month == 12)
    return 31;

  if (month == 4 || month == 6 || month == 9 || month == 11)
    return 30;

  if (month == 2) return isLeapYear(year) ? 29 : 28;

  return 0; // If month is incorrect
}

// Determine if it is a leap year
bool isLeapYear(int year)
{
  return year % 400 == 0 || (year % 4 == 0 && year % 100 != 0);
}
```

```
// Get the English name for the month
void printMonthName(int month)
{
 switch (month)
 {
  case 1:
   cout << "January";
   break;
  case 2:
   cout << "February";
   break;
  case 3:
   cout << "March";
   break;
  case 4:
   cout << "April";
   break;
  case 5:
   cout << "May";
   break;
  case 6:
   cout << "June";
   break;
  case 7:
   cout << "July";
   break;
  case 8:
   cout << "August";
   break;
  case 9:
   cout << "September";
   break;
  case 10:
   cout << "October";
   break;
  case 11:
   cout << "November";
   break;
  case 12:
   cout << "December";
 }
}
```

```
// Print month body
void printMonthBody(int year, int month)
{
 // Get start day of the week for the first date in the month
 int startDay = getStartDay(year, month);

 // Get number of days in the month
 int numberOfDaysInMonth = getNumberOfDaysInMonth(year, month);

 // Pad space before the first day of the month
 int i = 0;
 for (i = 0; i < startDay; i++)
  cout << "    ";

 for (i = 1; i <= numberOfDaysInMonth; i++)
 {
  cout << setw(4) << i;

  if ((i + startDay) % 7 == 0)
   cout << endl;
 }
}
```

```
// Get the start day of the first day in a month
int getStartDay(int year, int month)
{
 // Get total number of days since 1//1//1800
 int startDay1800 = 3;
 int totalNumberOfDays = getTotalNumberOfDays(year, month);

 // Return the start day
 return (totalNumberOfDays + startDay1800) % 7;
}

// Get the total number of days since January 1, 1800
int getTotalNumberOfDays(int year, int month)
{
 int total = 0;

 // Get the total days from 1800 to year - 1
 for (int i = 1800; i < year; i++)
  if (isLeapYear(i))
   total = total + 366;
  else
   total = total + 365;

 // Add days from Jan to the month prior to the calendar month
 for (int i = 1; i < month; i++)
  total = total + getNumberOfDaysInMonth(year, i);

 return total;
}
```



THANK YOU

VISIT US

WWW.XJTLU.EDU.CN

FOLLOW US

Xi'an Jiaotong-Liverpool University
西交利物浦大学