Lab 2

Lab 2 对应 lecture notes 的 Lecture 2（数据表示），训练目标是深刻理解 C++各种数据类型的二进制表示、变量、常量的基本用法，并利用这些知识编写简单的 C++程序。

表 2.1 C++的基本数据类型

基本数据类型	表示形式	C++中相应的 类型标识符	占用空间 （字节）
逻辑型	逻辑型	bool	1
字符型	字符型	char	1
		short	2
		int	4
		long	4
		long long	8
		unsigned short	2
		unsigned int	4
	单精度浮点型	float	4
	双精度浮点型	double	8

Problem 1.

Lecture 2 中我们学习了十进制数到其他进制的转换，通过编写下面的程序检验你的转换结果是否正确。要求：输入 3 个整数，分别是 10 进制数，16 进制数，8 进制数，将这 3 个数字分别以 10 进制，16 进制，8 进制的格式输出（数字用空格隔开，分 3 行）。

例子：

输入：

3　0x1a　014

输出：

3 3 3

26 1a 32

12 c 14

提示：（1）C++中 0x 开头的数字表示 16 进制数，0 开头的数字表示 8 进制数；（2）假设 x 为 int 类型变量，cin>>hex>>x;可以实现从键盘上读入 1 个 16 进制数，cin>>oct>>x;可以实现从键盘上读入 1 个 8 进制数，cout<<hex<<x;将以十六进制输出 x，cout<<oct<<x;将以

8 进制输出 x, cin>>dec>>x；可以实现从键盘上读入 1 个十进制数，cout<<dec<<x;可以实现以十进制输出 1 个十进制数。

Problem 2.

C++提供的函数 sizeof()可以得到数据类型或者变量所占内存的大小（字节数），例如，sizeof(int)的结果是 4，sizeof(double)的结果是 8。sizeof 同样可以作用于变量，例如，定义 double 类型的变量 x，那么 sizeof(x)的结果也是 8（即 double 类型的所占字节数）。编写程序，使用 sizeof()函数验证表 2.1 中所列出的所有数据类型的大小（string 类型除外）。

Problem 3.

Lecture 2 中我们学习了 C++各种数据类型的二进制表示方法，编写程序，验证 int, unsigned int, char, float 类型的二进制表示方法。要求：分别定义 int，unsigned int，char，float 类型变量，从键盘输入整数，整数，字符，浮点数赋值给对应的变量，输出这些变量的二进制形式。

例子：

输入：

-100　100　A　0.75

输出：

11111111111111111111111110011100

00000000000000000000000001100100

01000001

00111111010000000000000000000000

提示：(1) 程序前面加#include <bitset> (2) 对于 visual studio 2010，如果 x 是 int, unsigned int, float 类型的变量，可以使用 cout<<bitset<32>(*(int *)&a); 输出 x 的二进制；(3) 如果 x 是 char 类型的变量，可以使用 cout<<bitset<8>(*(unsigned char*)&c)<<endl;输出 x 的二进制；

Problem 4.

我们知道，unsigned int 类型只能表示非负整数，范围是 0~4294967295，但有的同学编程的时候忽略了这些限制，发生了意想不到的事情。编写程序，验证下面两件事情：(1) 定义一个 unsigned int 变量，给这个变量赋值为负整数，会发生什么？(2) 定义两个 unsigned int 类型的变量，让这两个变量的和超过 4294967295，此时会发生什么？请用学过的知识对发现的现象进行解释。

Problem 5.

我们知道，int 类型能表示的范围是-2147483648~2147483647。如果两个很大的 int 类型正整数相加（和超过 2147483647）会发生正溢出，两个很大的 int 类型负整数相加（和小于-2147483648）会发生负溢出，请编写程序，观察正溢出和负溢出发生时的现象，利用学过的知识对发现的现象进行解释。

Problem 6.

有位同学写了一个程序，定义了一个 float 类型的变量 x，初始化为 0，然后每次给 x 增加 0.1，一共迭代了 4000000 次，然后输出 x。他期望能够得到 400000，但他惊讶的发现最终的结果为 384525。请复现他的试验，并根据学过的知识对发生的现象进行解释。

提示：假设 x 为 float 类型的变量，初始值为 0，那么迭代 N 次、每次 x 增加 0.1 的程序实现如下：

```
float x = 0;
for(int i = 0; i < N; i ++) {
    x = x + 0.1;
}
```

Problem 7.

数据类型转换对于 C++初学者来说非常容易引起错误，请验证下面这些代码片段的结果，并利用学过的知识对结果进行解释。

```
int a;
a = 3.14;
cout<<a<<endl;

float a;
a = 3;
cout<<a<<endl;

float a;
double b = 3.14;
a = b;
cout<<a<<endl;

double a;
float b = 3.14;
a = b;
cout<<a<<endl;

int a;
a = 'A';
cout<<a<<endl;
cout<<a+1<<endl;

unsigned int a;
a = -3;
cout<<a<<endl;

char a;
a = 97;
cout<<a<<endl;
```

```
cout<<a+1<<endl;

unsigned int u = 10;
int i = -42;
cout<<i + i<<endl;
cout<<u + i<<endl;
```