实验 #1
截断误差与舍入误差

刘扬 2011012162

实验目标

通过实验的过程了解两种基本误差：截断误差与舍入误差

实验原理

截断误差

对于一个数学模型如果我们使用数值方法进行求解，我们必须放弃一定的精度，从而产生了截断误差。
我们只能通过使用精度更高的算法

舍入误差

来缩小截断误差。 然而在计算机上面实现数值方法的时候我们不能保证所有的运算就像理想的那样，
只有截断误差，由于我们使用的数据类型不同，导致我们没有办法存下实数，尤其是无理数的所有小数位数，
这时候就产生了舍入误差。这也是无法避免的。

问题中N的估算

根据问题中给出的误差公式$\|x_n - ln2\| \leq \frac{1}{n+1}$我们对于不同的误差进行估算:

当$e_1 = \frac{1}{2} \times 10^{-5}$，所以有$\frac{1}{n+1} \leq \frac{1}{2} \times 10^{-5}$,推算得$n \geq 199999$

当$e_2 = \frac{1}{2} \times 10^{-6}$，所以有$\frac{1}{n+1} \leq \frac{1}{2} \times 10^{-6}$,推算得$n \geq 1999999$

算法设计与实现

由于需要保证运算精度，而python没有单精度型，所以使用了C++进行程序设计，根据公式
$x_n = \sum_{k=1}^{n} \frac{(-1)^{k-1}}{k}$进行计算，得到结果

实验结果

Listing 1: 实验结果

```
e = 0.5e-5
Answer: 0.69315219
```

```
Err : 0.000004999
N : 40089
e = 0.5e-6
Answer: 0.69314766
Err : 0.000000469
N : 58765
```

体会与问题

通过观察程序的输出，我们发现实际上需要的循环次数比理论上的估计值小很多，我觉得这是因为计算机在运算的时候，由于表示精度的差别造成了舍入，从而抹去了一些末尾的结果，导致我们可以通过更少的循环次数获得解。
比如求解在计算$\frac{1}{3}$的时候,我们实际上只用了几位，即用0.333333 表示了$\frac{1}{3}$ 导致了结果最后有偏差,
正是舍入误差导致了最后实际上的n和理论上计算的n不同。

附:实验代码

Listing 2: 实验的代码

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

const double stand = 0.693147190546;

int get_n(float e){
     int n = 0;
     float ans = 0.0;
     int i = 0;
     int f = -1;
     while (abs(ans - stand) > e){
          i++;
          f = f * -1;
          ans = ans + (1.0/(float)(i)) * (float)(f);
     }
     //std::cout << "Answer : " << ans << std::endl;
     printf("Answer: %.8f\n",ans);
     printf("Err : %.9f\n",abs(ans - stand));
     return i;
}

int main(){
     std::cout << "e = 0.5e-5" << std::endl;
     int n1 = get_n(0.5 * 0.00001);
     std::cout << "N : " << n1<< std::endl;
     std::cout << "e = 0.5e-6" << std::endl;
     int n2 = get_n(0.5 * 0.000001);
     std::cout << "N : " << n2 << std::endl;
     return 0;
}
```