北京航空航天大学数学科学学院实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 课程名称：科学计算通识实验课 | | 实验名称：实验一： C++基础操作、误差影响基础实验 | |
| 实验类型： 演示性实验□ 验证性实验☑ 综合性实验□ 设计性实验□ | | | |
| 班级：暑期班 | 姓名：涂磊 | | 学号：22377211 |
| 实验日期： 2024/7/1 | 指导教师：冯成亮 | | 实验成绩： |
| 实验环境：（所用仪器设备及软件）  Windows + VS-code, Ubuntu 20.04.6 + g++ | | | |
| 实验目的与实验内容：  【目的要求】  通过本实验使学生熟悉个人电脑上C++代码的编写与调试，服务器上的代码编译与运行； 学会服务器上的一些linux基本命令；了解科学计算过程中舍入误差的产生与积累过程，领会算法设计原则对应的误差影响内涵及其必要性，初步体验迭代算法的稳定性差异；  【实验内容】  实验1.1：（比较固定位有效数字四舍五入所导致的误差影响）  分别计算  f1 = 0.12346845;  f2 = 0.012346845;  f3 = 0.0012346845;  f4 = 0.0000012346845;  的四位有效数字近似值；计算它们的绝对误差和相对误差；  实验1.2：（体验有效数字位数四舍五入的二进制陷阱）  计算 f5 = 4.015 \* 100 的 3位有效数字近似值，比较该近似值与理论近似值得差别；  计算 浮点数f6 = 7654321.0f与f7 = 0.4f的和，以及它们和的7位有效数字近似值；比较该近似值与理论近似值的差别；  实验2.1：（比较相近大小两个数值作差的误差影响）  对双精度数 pi0 = 3.141592653589793 与 pi1 = pi0 + 0.000123456789123 做8位有效数字近似 ppi0 与 ppi1，分别计算它们的绝对误差与相对误差；  计算两个近似值ppi0 与 ppi1 的差的 8位有效数字近似值 mpp，计算该近似差值的绝对误差与相对误差；  实验2.2：（比较大数与小数求和的误差影响）  对浮点数pi20 = 7654321.0f 与 5个浮点数pi21 = 0.2f顺序求和；  将5个浮点数pi21 = 0.2f顺序求和后与浮点数pi20 = 7654321.0f 求和；  比较交换计算顺序后的结果差异；  实验2.3：（比较小数被除时的误差影响）  对双精度数pi30 = 3.141592653589793 与 pi31 = 0.000123456789123 做 8位有效数字近似 ppi30 与 ppi31，分别计算它们的绝对误差与相对误差；  计算两个近似值ppi30 与 ppi31比值的8位有效数字近似值 mpp30， 计算该近似比值的绝对误差和相对误差；  实验3.1（体验算法的稳定性差异对计算结果的影响）  对分别采用  与  的迭代算法做n=0 to 9 的4位有效数字计算；输出每一步的值，并与精确结果做比较，输出其 绝对误差与相对误差表。 | | | |
| 实验过程与结果：  实验1.1：（比较固定位有效数字四舍五入所导致的误差影响）  #include <stdio.h>  #include "convert.hpp"  #include "error.hpp"  int main(void) {      double f1 = 0.12346845;      double f2 = 0.012346845;      double f3 = 0.0012346845;      double f4 = 0.00012346845;      double f1\_4 = con\_d\_4(f1);      double f2\_4 = con\_d\_4(f2);      double f3\_4 = con\_d\_4(f3);      double f4\_4 = con\_d\_4(f4);      printf("real\tobservation\tabsolute\_error\trelative\_error\n");      printf("%f\t%f\t%.8f\t%.8f\n",f1, f1\_4, absolute\_error(f1\_4, f1), relative\_error(f1\_4, f1));      printf("%f\t%f\t%.8f\t%.8f\n",f2, f2\_4, absolute\_error(f2\_4, f2), relative\_error(f2\_4, f2));      printf("%f\t%f\t%.8f\t%.8f\n",f3, f3\_4, absolute\_error(f3\_4, f3), relative\_error(f3\_4, f3));      printf("%f\t%f\t%.8f\t%.8f\n",f4, f4\_4, absolute\_error(f4\_4, f4), relative\_error(f4\_4, f4));      //printf("%f 的四位有效数字近似值的绝对误差为 %.8f,相对误差为 %.8f\n", f1, absolute\_error(f1\_4, f1), relative\_error(f1\_4, f1));      //printf("%f 的四位有效数字近似值的绝对误差为 %.8f,相对误差为 %.8f\n", f2, absolute\_error(f2\_4, f2), relative\_error(f2\_4, f2));      //printf("%f 的四位有效数字近似值的绝对误差为 %.8f,相对误差为 %.8f\n", f3, absolute\_error(f3\_4, f3), relative\_error(f3\_4, f3));      //printf("%f 的四位有效数字近似值的绝对误差为 %.8f,相对误差为 %.8f\n", f4, absolute\_error(f4\_4, f4), relative\_error(f4\_4, f4));      return 0;  }    实验2.1：（比较相近大小两个数值作差的误差影响）  #include <stdio.h>  #include "convert.hpp"  #include "error.hpp"  int main(void) {      double pi0 = 3.141592653589793;      double pi1 = pi0 + 0.000123456789123;      double ppi0 = con\_d\_8(pi0);      double ppi1 = con\_d\_8(pi1);      printf("Approximate to eight significant figures:\n");      printf("real\tobservation\tabsolute\_error\trelative\_error\n");      printf("%lf\t%lf\t%.16lf\t%.16lf\n", pi0, ppi0, absolute\_error(ppi0, pi0), relative\_error(ppi0, pi0));      printf("%lf\t%lf\t%.16lf\t%.16lf\n", pi1, ppi1, absolute\_error(ppi1, pi1), relative\_error(ppi1, pi1));      //printf("%lf 做八位有效数字近似的绝对误差为 %.16lf, 相对误差为 %.16lf\n", pi0, absolute\_error(ppi0, pi0), relative\_error(ppi0, pi0));      //printf("%lf 做八位有效数字近似的绝对误差为 %.16lf, 相对误差为 %.16lf\n", pi1, absolute\_error(ppi1, pi1), relative\_error(ppi1, pi1));      double mp = con\_d\_8(ppi0 - ppi1);      printf("\n\n");      printf("The difference between the two approximations\n");      printf("real\tobservation\tabsolute\_error\trelative\_error\n");      printf("%lf\t%lf\t%.16lf\t%.16lf\n", 0.000123456789123, mp, absolute\_error(mp, 0.000123456789123), relative\_error(mp, 0.000123456789123));      //printf("近似差值%lf 的绝对误差为 %.16lf, 相对误差为 %.16lf\n", mp, absolute\_error(mp, 0.000123456789123), relative\_error(mp, 0.000123456789123));      return 0;  }    实验2.2：（比较大数与小数求和的误差影响）  #include <stdio.h>  #include "convert.hpp"  #include "error.hpp"  int main(void) {      float pi20 = 7654321.0f;      float pi21 = 0.2f;      float result1 = pi20;      for(int i = 0; i < 5; i++){          result1 += pi21;      }      float result2 = pi20;      float cd = pi21;      for (int i = 0; i < 4; i++){          cd += pi21;      }      result2 += cd;      //printf("The sequential sum of %f and five %fs results in %f\n", pi20, pi21, result1);      //printf("The sum of %f and five %fs results in %f\n", pi20, pi21, result2);      printf("We believe that after summing the five decimals first, then adding the large number is the true value\n\n");      printf("real\tobservation\tabsolute\_error\trelative\_error\n");      printf("%f\t%f\t\t%.8lf\t\t%.8lf\n", result2,result1,absolute\_error(result1,result2),relative\_error(result1,result2));      //printf("绝对误差为%.8lf,相对误差为%.8lf\n", absolute\_error(result1,result2), relative\_error(result1,result2));      return 0;  }    实验2.3：（比较小数被除时的误差影响）  #include <stdio.h>  #include "convert.hpp"  #include "error.hpp"  int main(void) {      double pi30 = 3.141592653589793;      double pi31 = 0.000123456789123;      double ppi30 = con\_d\_8(pi30);      double ppi31 = con\_d\_8(pi31);      printf("The difference between eight significant figures and the original number is:\n");      printf("real\tobservation\tabsolute\_error\trelative\_error\n");      printf("%lf\t%lf\t%.16lf\t%.16f%%\n", pi30, ppi30, absolute\_error(ppi30,pi30), relative\_error(ppi30,pi30));      printf("%lf\t%lf\t%.16lf\t%.16f%%\n", pi31, ppi31, absolute\_error(ppi31,pi31), relative\_error(ppi31,pi31));      //printf("%lf 的八位有效数字近似的绝对误差为%.16lf,相对误差为%.16f%%\n", pi30, absolute\_error(ppi30,pi30), relative\_error(ppi30,pi30));      //printf("%lf 的八位有效数字近似的绝对误差为%.16lf,相对误差为%.16f%%\n", pi31, absolute\_error(ppi31,pi31), relative\_error(ppi31,pi31));      double mp30 = con\_d\_8(ppi30)/con\_d\_8(ppi31);      double mpp30 = con\_d\_8(mp30);      double real = pi30/pi31;      printf("\n\n");      printf("The difference between the eight significant figures of the ratio and the original ratio is:\n" );      printf("real\tobservation\tabsolute\_error\trelative\_error\n");      printf("%lf\t%lf\t%.16lf\t%.16f\n", real, mpp30, absolute\_error(mpp30,real), relative\_error(mpp30,real));      //printf("%lf与%lf比值 %lf 的 八位有效数字近似的 绝对误差为%.16lf,相对误差为%.16lf%%\n", ppi30, ppi31, mp30,absolute\_error(mpp30,real), relative\_error(mpp30,real));      return 0;  }    实验3.1（体验算法的稳定性差异对计算结果的影响）  #include <stdio.h>  #include <math.h>  #include "convert.hpp"  #include "error.hpp"  //第一种迭代方法  double func1(double I0,int n){      double I1 = 1 - I0 \* n;      return I1;  }  //第二种迭代方法  double func2(double I1, int n){      double I0 = (double)(1/(double)n) \* (double)(1 - I1);      return I0;  }  int main(void) {      //真实值      //printf("真实值：\n\n");      double e = exp(1);      double I0 = 1 - 1 / e;      double I1 = 1/e;      double I2 = 1 - 2/e;      double I3 = -2 + 6/e;      double I4 = 9 - 24/e;      double I5 = -44 + 120/e;      double I6 = 265 - 720/e;      double I7 = -1854 + 5040/e;      double I8 = 14833 - 40320/e;      double I9 = -133496 + 362880/e;      double I\_0[10] = {I0, I1, I2, I3, I4, I5, I6, I7, I8, I9};      for(int i = 0; i < 10; i++) {          printf("I%d = %lf\n", i, I\_0[i]);      }        //printf("\n");  //第一种迭代方法      //printf("第一种迭代方法：\n\n");      double I\_1[10];      I\_1[0] = I0;        for(int i = 0; i < 10; i++) {          I\_1[i+1] = con\_d\_4(func1(I\_1[i], i+1));      }      //for(int i = 0; i < 10; i++) {      //    printf("I%d = %.5lf\n其与真值的绝对误差为%.16lf,相对误差为：%.16lf\n", i, I\_1[i], absolute\_error(I\_1[i], I\_0[i]), relative\_error(I\_1[i], I\_0[i]));      //}      //printf("\n");  //第二种迭代方法      //printf("第二种迭代方法：\n\n");      double I\_2[10];      I\_2[9] = I9;      for(int i = 9; i > 0; i--) {          I\_2[i-1] = con\_d\_4(func2(I\_2[i], i));      }      //for(int i = 0; i < 10; i++) {      //    printf("I%d = %.5lf\n其与真值的绝对误差为%.16lf,相对误差为：%.16lf\n", i, I\_2[i], absolute\_error(I\_2[i], I\_0[i]), relative\_error(I\_2[i], I\_0[i]));      //}        printf("--------------------\n");      printf("Index\trealvalue\tobserve\_1\tabsolute\_error\_1\trelative\_error\_1\t\tobserve\_2\tabsolute\_error\_2\trelative\_error\_2\n");      for(int i = 0; i < 10; i++) {          printf("I%d\t%.4lf\t\t%.4lf\t\t%.16lf\t%.16lf\t\t%.4lf\t\t%.16lf\t%.16lf\n",i, I\_0[i], I\_1[i], absolute\_error(I\_1[i], I\_0[i]), relative\_error(I\_1[i], I\_0[i]), I\_2[i], absolute\_error(I\_2[i], I\_0[i]), relative\_error(I\_2[i], I\_0[i]));      }      return 0;  } | | | |
| 实验分析与总结：  经过本次实验，了解到了误差产生的原因以及为什么要避免误差，如何避免误差。  强化了编程能力，学会了如何使用远程服务器辅助完成代码的运行。 | | | |

注：若填写内容较多，可在背面继续填写。