SouthWest University

Lab report

Couse name C programming

Semester 2019 - 2020 - 2

Grade 2019 class Software Engineering 3

student Reg. No 222019321062074

Tutor James Chen

School of Computer and Information Science

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Lab 3 | | | **Practicing on functions** | | | | | |
| Issue Date | | 2020年3月30日 | | | experimental types | | □validation experiment,  □comprehensive experiment  ☑design experiment | |
| **Goal**  • You will practice using functions  **Assignment**  1.Write a program that can take three integers as the sides of a triangle and  output the circumference and area of this triangle. (use two user-defined functions to implement the calculation of circumference and area of a triangle separately.  2 Write a program that can find the root of the equation ax2+bx+c=0. (using at least one user-defined functions to complete the task.)  3.Write a function lcm() that takes two integer arguments and returns their lcm. The lcm() function should calculate the least common multiple by calling the gcd() function. gcd is a function that can take two integer arguments and return their greatest common divisor.  lcm(u,v)=uv/gcd(u,v) u,v>=0  4. Write a program to output the first n numbers of a Fibonacci series by using recursive calling. | | | | | | | | |
| * experimental contents and process  1. Analyze what the problems say, and understand what it means. 2. Clear object procedure. 3. Compilation object process. 4. Make the program functional. | | | | | | | | |
| * screen shots of the VC++ or other IDE for C(Dev c++，VS2010. etc)showing the output results of running your Lab code.   Ex1：    Ex2：    Ex3：    Ex4： | | | | | | | | |
| * Lab Code   Ex1：  #include <stdio.h>  #define PI 3.14159265358979323846  double area(double r)  {  return r \* r \* PI;  }  double circum(double r)  {  return 2 \* PI \* r;  }  int main (void)  {  double r;  scanf("%lf", &r);  printf("circumference:%.2lf\n", circum(r));  printf("area:%.2lf", area(r));  return 0;  }  Ex2：  #include <stdio.h>  #include <math.h>  int judge(double x)  {  if (fabs(x) - 1E-8 <= 0)  return 1;  return 0;  }  double calc(double a, double b, double c)  {  double delta;  delta = b \* b - 4 \* a \* c;  if (judge(a) && judge(b) && judge(c))  {  printf("Zero Equation");  return 0;  }  if (judge(a) && judge(b))  {  printf("Not An Equation");  return 0;  }  if (judge(a))  {  printf("%.2lf", -c / b );  return 0;  }  if (judge(delta))  {  printf("%.2lf", -b / a / 2);  return 0;  }  if (delta >= 0)  {  printf("%.2lf\n%.2lf", (-b + sqrt(delta)) / a / 2, (-b - sqrt(delta)) / a / 2);  return 0;  }  if (judge(b))  {  printf("0.00+%.2lfi\n0.00-%.2lfi\n", sqrt(-delta) / a / 2, sqrt(-delta) / a / 2);  return 0;  }  printf("%.2lf+%.2lfi\n", -b / a / 2, sqrt(-delta) / a / 2);  printf("%.2lf-%.2lfi\n", -b / a / 2, sqrt(-delta) / a / 2);  return 0;  }  int main (void)  {  double a, b, c;  scanf("%lf%lf%lf", &a, &b, &c);  calc(a, b, c);  return 0;  }  Ex3：  #include <stdio.h>  int gcd(int x, int y)  {  if (y)  return gcd(y, x%y);  else  return x;  }  int lcm(int x, int y)  {  return x \* y / gcd(x, y);  }  int main (void)  {  int m, n;  scanf("%d%d", &m, &n);  printf("%d", lcm(m, n));    return 0;  }  Ex4：  #include <stdio.h>  #define ll long long  ll fibo(ll n)  {  if (n == 1 || n == 2)  return 1;  else  return fibo(n - 1) + fibo(n - 2);  }  int main (void)  {  ll a;  scanf("%lld", &a);  printf("%lld", fibo(a));    return 0;  } | | | | | | | | |
| * experimental summary/Analysis   Through this experiment, I got a better understand of using functions, properly use of functions can divide a complex program into small pieces to make it more comprehensible, and help people better handle different problems.  In ex 2 and 3, I used 2 user-defined functions, let the main function to call one, and then the called one to call the other one.  In ex4, I knew about recursive calling, which means a function calls itself, it helps to reduce the amount of code, and makes the program more legible. | | | | | | | | |
|  | Criteria | | | | | | | scale |
| Goal | | | | | | | A B C D E |
| Process | | | | | | |
| Design | | | | | | |
| Algorithm | | | | | | |
| Code | | | | | | |
| Data/Results | | | | | | |
| summary | | | | | | |
| written | | | | | | |
| Score | | |  | | tutor Signature：  Date: : | | |
| * Lab Evaluation Criteria   A: This lab is exceptional, working and meeting all of the specifications.The code is exceptionally well organized and very easy to follow.The code could be reused as a whole or each routine could be reused.The documentation is well written and clearly explains what the code is accomplishing and how.The program was delivered on time.The code is extremely efficient without sacrificing readability and understanding.  B: This lab is very good--works and produces the correct results and displays them correctly. It also meets most of the other specifications. The code is fairly easy to read. Most of the code could be reused in other programs. The documentation consists of embedded comment and some simple header documentation that is somewhat useful in understanding the code. The program was delivered within a week of the due date. The code is fairly efficient without sacrificing readability and understanding.  C: This lab is adequate, with only minor deficiencies. The program produces correct results but does not display them correctly. The code is readable only by someone who knows what it is supposed to be doing. Some parts of the code could be reused in other programs. The documentation is simply comments embedded in the code with some simple header comments separating routines. The code was within 2 weeks of the due date. The code is brute force and unnecessarily long..  D: This lab shows some effort but has at least one major deficiency.The program is producing incorrect results. The code is poorly organized and very difficult to read. The code is not organized for reusability. The documentation is simply comments embedded in the code and does not help the reader understand the code. The code was more than 2 weeks overdue. The code is huge and appears to be patched together.  E: This lab is poorly written and shows very little effort or understanding. | | | | | | | | |