## 毕业设计电算程序说明

一．程序功能：

1. 校核齿轮弯曲强度，接触强度；
2. 校核轴在齿轮的作用下的挠度和强度。

二．程序流程图：

开始

打开数据文本文件

读取齿轮数

齿轮数>0?

读取齿轮数据

生成齿轮，

齿轮数-1

校核所有齿轮弯曲强度，接触强度，输出结果

否

是

读取轴数据

轴上齿轮数量>0?

读取齿轮id和a

校核轴的挠度和强度，齿轮数-1，输出结果

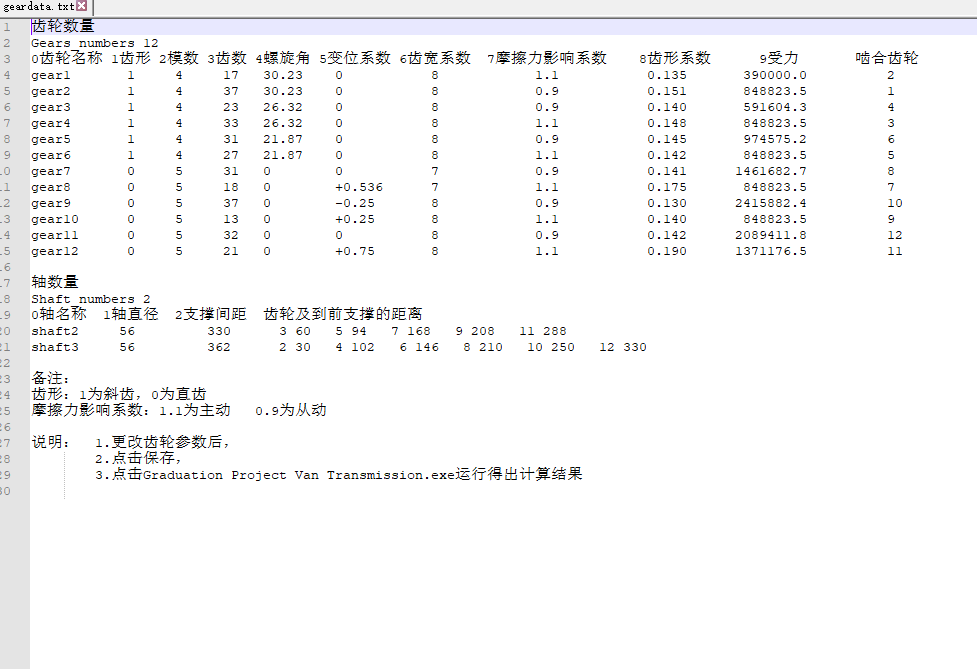
结束

是

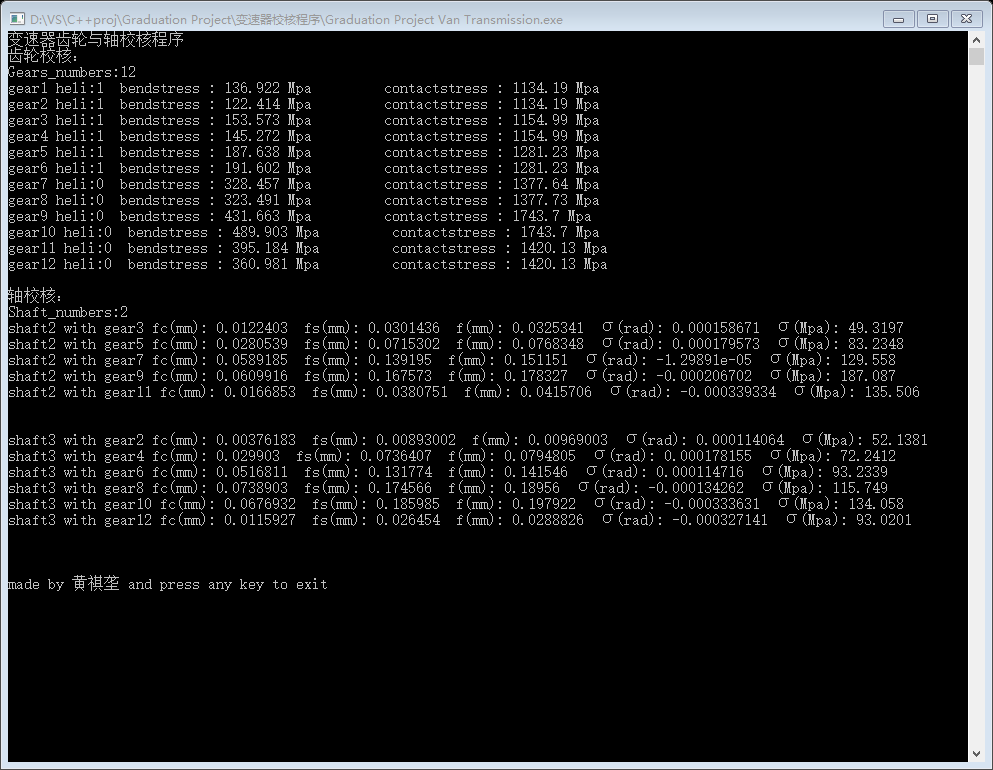
否

三．数据的输入：

齿轮数据和轴数据的输入通过数据文档：geardata.txt文档来传递



四．程序运行结果：



五．程序源码：

1.主程序

// Graduation Project Van Transmission.cpp : 定义控制台应用程序的入口点。

#include <iostream>

#include <fstream>

#include <sstream>

#include <vector>

#include <list>

#include "stdafx.h"

#include "Gear.h"

#include "Shaft.h"

#define \_USE\_MATH\_DEFINES

#include <math.h>

//"D:\\VS\\C++proj\\Graduation Project\\Graduation Project Van Transmission"

using namespace std;

int main()

{

cout << "变速器齿轮与轴校核程序" << endl;

string data = "geardata.txt";

fstream geardata;

geardata.open(data);

if (!geardata)

{

cerr << "open file err" << endl;

cout << "press any key to exit" << endl;

return getchar();

}

cout << "齿轮校核：" << endl;

int gearnumbers,shaftnumbers;

vector<Gear> gears;

vector<Shaft> shafts;

string temp,word;

getline(geardata, temp);

getline(geardata, temp);

istringstream firstline(temp);

firstline >> word;

if (word != "Gears\_numbers")

{

cerr << "Gear\_numbers err" << endl;

cout << "press any key to exit" << endl;

return getchar();

}

firstline >> word; gearnumbers = stoi(word);

cout << "Gears\_numbers:"<< gearnumbers << endl;

getline(geardata, temp);

for (int i = 0; i < gearnumbers; i++)

{

getline(geardata, temp);

istringstream gear(temp);

vector<string> vs;

while (gear >> word)

vs.push\_back(word);

if (vs.size() != 11)

{

cerr << "Gear "<<i+1<<" parameter numbers err" << endl;

cout << "press any key to exit" << endl;

return getchar();

}

Gear geartemp(vs[0].c\_str(), stoi(vs[2]), stoi(vs[3]), (bool)stoi(vs[1]), stod(vs[4]), stod(vs[5]));

geartemp.setparameters(stod(vs[9]), stod(vs[7]), stod(vs[6]), stod(vs[8]), stoi(vs[10]));

gears.push\_back(geartemp);

}

for (int i = 0; i < gearnumbers; i++)

{

gears[i].outbendstress(); gears[i].outcontactstress(gears[gears[i].mateid - 1]);

}

cout << "\n轴校核：" << endl;

while (getline(geardata, temp))

{

firstline.clear(); firstline.str(temp);

firstline >> word;

if (word == "Shaft\_numbers")

break;

}

if (geardata.fail())

{

cerr << "\nend of the file" << endl;

cout << "press any key to exit" << endl;

return getchar();

}

firstline >> word; shaftnumbers = stoi(word);

cout << "Shaft\_numbers:" << shaftnumbers << endl;

getline(geardata, temp);

for (int i = 0; i < shaftnumbers; i++)

{

getline(geardata, temp);

istringstream shaft(temp);

vector<string> vs;

while (shaft >> word)

vs.push\_back(word);

if ((vs.size() < 3)|| (vs.size()%2==0))

{

cerr << "Shaft " << i + 1 << " parameter numbers err" << endl;

cout << "press any key to exit" << endl;

return getchar();

}

Shaft shafttemp(vs[0], stod(vs[1]), stod(vs[2]));

for (int j = 3; j <= vs.size() - 2; j += 2)

{

shafttemp.deflection\_strength(gears[stod(vs[j])-1], gears[gears[stod(vs[j])-1].mateid - 1], stod(vs[j+1]));

}

cout << "\n" << endl;

}

2．齿轮程序

#include "stdafx.h"

#include "Gear.h"

double Gear::bendstress()

{

double bs;

if(helical)

bs = 2 \* tmax\*cos(beyta\*M\_PI/180)\*ko / 3.1415926 / pow(module, 3) / z / kc / y/ 2.0;

else

bs = 2 \* tmax\*ko\*kf / 3.1415926 / pow(module, 3) / z / kc / y;

return bs;

}

double Gear::contactstress(const Gear &mgear)

{

double alfa = 20.0 \* M\_PI / 180;

double a = module\*(z + mgear.z) / 2;

double xalfa = ainv(2 \* (xigma+mgear.xigma)\*tan(alfa) / (z + mgear.z) + inv(alfa));

double bey = beyta\*M\_PI / 180.0;

double F = 2 \* tmax / (module\*z)/(cos(xalfa)\*cos(bey));

double rz = kf == 1.1 ? module\*z / cos(bey) / 2 : module\*mgear.z / cos(bey) / 2;

double rb = !(kf == 1.1) ? module\*z / cos(bey) / 2 : module\*mgear.z / cos(bey) / 2;

double roz = rz\*sin(xalfa) / pow(cos(bey), 2);

double rob = rb\*sin(xalfa) / pow(cos(bey), 2);

double cs = 0.418\*sqrt(F\*elastic/b\*(1/roz+1/rob));

return cs;

}

void Gear::solveftfrfn(const Gear &mgear, double &ft, double &fr, double &fn)const

{

double alfa = 20.0 \* M\_PI / 180;

double a = module\*(z + mgear.z) / 2;

double xalfa = ainv(2 \* (xigma + mgear.xigma)\*tan(alfa) / (z + mgear.z) + inv(alfa));

double bey = beyta\*M\_PI / 180.0;

ft = 2 \* tmax / (module\*z/ cos(bey));

fr = ft\*tan(xalfa) / cos(bey);

fn = ft / (cos(xalfa)\*cos(bey));

}

double inv(const double &x)

{

return tan(x) - x;

}

double ainv(const double &x)

{

double min = 0, max = M\_PI / 2, y = (min + max) / 2;

while (abs(inv(y) - x)>0.000000001)

{

y = (min + max) / 2;

if (inv(y) > x)

max = y;

else

min = y;

}

return y;

}

double get\_xigma\_a(double m, double z1, double z2, double xigma)

{

double alfa = 20.0 \* M\_PI / 180;

double a = m\*(z1 + z2) / 2;

double xalfa = ainv(2 \* xigma\*tan(alfa) / (z1 + z2) + inv(alfa));

double xa = a\*cos(alfa) / cos(xalfa);

return xa;

}

double get\_xigma(double m, double z1, double z2, double xa)

{

double alfa = 20.0 \* M\_PI / 180;

double a = m\*(z1 + z2) / 2;

double xalfa = acos(a\*cos(alfa) / xa);

double xigma = (z1 + z2)\*(inv(xalfa) - inv(alfa)) / 2 / tan(alfa);

return xigma;

}

float totalz(float b, float A, float m)

{

return A \* 2 \* cos(b\*M\_PI / 180) / m;

}

float solvebeyta(float z, float A, float m)

{

return acos(m\*z / 2.0 / A) \* 180 / M\_PI;

}

float optbeytan(float z1, float z2, float zl, float zh, float b)

{

float n = z2 / (z1 + z2)\*(1 + zh / zl);

return atan(tan(b\*M\_PI / 180) / n) \* 180 / M\_PI;

}

3．轴程序

#include "stdafx.h"

#include "Gear.h"

#include "Shaft.h"

void Shaft::deflection\_strength(const Gear &gear1, const Gear &gear2, float a)

{

double ft = 0, fr = 0, fn = 0;

double b = length - a;

gear1.solveftfrfn(gear2, ft, fr, fn);

double fc = fr\*pow(a, 2)\*pow(b, 2) / (3 \* elastic\*M\_PI\*pow(d, 4) / 64 \* length);

double fs = ft\*pow(a, 2)\*pow(b, 2) / (3 \* elastic\*M\_PI\*pow(d, 4) / 64 \* length);

double f = sqrt(fc\*fc + fs\*fs);

double xig = fr\*a\*b\*(b - a) / (3 \* elastic\*M\_PI\*pow(d, 4) / 64 \* length);

double Mc = fr\*a\*b / length;

double Ms = ft\*a\*b / length;

double Tn = gear1.tmax;

double M = sqrt(Mc\*Mc + Ms\*Ms + Tn\*Tn);

double lvmda = 32 \* M / M\_PI / pow(d, 3);

cout << id + " with " + gear1.name + " fc(mm): " << fc << " fs(mm): " << fs << " f(mm): " << f << " σ(rad): " << xig << " σ(Mpa): " << lvmda << endl;

}