第一题

#include <stdio.h>

#define N 10

int main()

{

double Higher90 = 0, Lower =0, Avg = 0, Score[N]; // 定义九十以上人数，平均分以下，平均分，成绩组

for (int i = 0; i <= N - 1;i++)

{

scanf\_s("%lf",&Score[i]); //输出成绩组

Avg += Score[i]; //记录总分

if (Score[i] > 90.0) Higher90++; //90以上人数加一

}

Avg = Avg / N; //计算平均分

printf("90分以上占比 %lf \n", (Higher90 / N) \* 100);

printf("平均分 %lf \n", Avg);

for (int i = 0; i <= N-1;i++)

{

if (Score[i] < Avg) Lower++; //计算平均分以下

}

printf("低于平均分占比 %lf \n", (Lower / N) \* 100);

return 0;

}

第二题

#include <stdio.h>

#define N 5

void BubbleSort(float Arr[], int Len) //冒泡排序

{

float Temp;

bool Flag;

for (int i = 0; i <= Len - 2; i++)

{

Flag = true;

for (int j = 0; j <= Len - 2 -i; j++)

{

if (Arr[j + 1] > Arr[j])

{

Temp = Arr[j];

Arr[j] = Arr[j + 1];

Arr[j + 1] = Temp;

Flag = false;

}

}

if (Flag == true) break;

}

}

int main()

{

float Score[N], Avg =0;

for (int i = 0; i <= N - 1; i++) //输入成绩

{

scanf\_s("%f", &Score[i]);

}

for (int i = 0; i <= N - 1; i++) //排除不合格成绩

{

while (Score[i] < 80.0 || Score[i] >100)

{

printf("%d 号评委不合理，重新输入：\n", i+1);

scanf\_s("%f", &Score[i]);

}

}

BubbleSort(Score, N); //冒泡排序

for (int i = 0; i <= N - 1; i++)

{

printf("%6.2f", Score[i]); //输出成绩

}

float SocreIn, NewSocre[N+1];

printf("\n 输入新成绩：\n"); //得到新成绩

scanf\_s("%f", &SocreIn);

while (SocreIn < 80.0 || SocreIn >100)

{

printf("不合理，重新输入：\n");

scanf\_s("%f", &SocreIn);

}

int k = N;

for (k = 0; k <= N - 1; k++) //寻找新成绩的位置

{

if (Score[k] <= SocreIn)

{

break;

}

}

NewSocre[k] = SocreIn; //插入并复制其余数组

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

{

NewSocre[i] = Score[i];

}

for (int i =k+1; i < N;i++)

{

NewSocre[i] = Score[i-1];

}

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

{

printf("%6.2f", NewSocre[i]); //输出成绩

Avg += NewSocre[i];

}

printf("\n 平均值: %6.2f", Avg /(float) (N + 1)); //输出平均成绩

return 0;

}

第三题

#include <stdio.h>

int main()

{

long FibNum[21] = {0,1,1,2}; //预设好斐波那契头部

for (int i = 4; i <= 20; i++)

{

FibNum[i] = FibNum[i - 1] + FibNum[i - 2]; //利用公式逐项求出

}

for (int i = 1; i <= 20; i++)

{

printf("%d ", FibNum[i]); //输出

}

return 0;

}

第四题

#include <stdio.h>

void BubbleSort(int Arr[], int Len) //冒泡排序

{

int Temp;

bool Flag;

for (int i = 0; i <= Len - 2; i++)

{

Flag = true;

for (int j = 0; j <= Len - 2 - i;j++)

{

if (Arr[j] < Arr[j + 1])

{

Temp = Arr[j];

Arr[j] = Arr[j + 1];

Arr[j + 1] = Temp;

Flag = false;

}

}

}

}

int main()

{

int ArrNum[8] ={0};

for (int i = 0;i < 8;i++)

{

scanf\_s("%d", &ArrNum[i]); //输入数列

}

BubbleSort(ArrNum, 8); //从大到小排序

if (ArrNum[0] == ArrNum[1]) //判定第一个数是否有重复

{

printf("%d ", ArrNum[0]);

}

for (int i = 1;i <= 6;i++)

{

if (ArrNum[i] == ArrNum[i + 1] && ArrNum[i] != ArrNum[i - 1]) //判定其余数是否是新数重复（和后一个的大，又不等于前一个）

{

printf("%d ", ArrNum[i]);

}

}

return 0;

}

第五题

#include <stdio.h>

#define N 3

int main()

{

int Arr[N][N];

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

{

for (int j = 0; j < N;j++)

{

scanf\_s("%d", &Arr[i][j]);

}

}

int max, mid, min;

for (int i = 0; i < N;i++) //处理每一行

{

max = Arr[i][0] > Arr[i][1] ? Arr[i][0] : Arr[i][1]; //得到一行中三个数（大中小）

max = max > Arr[i][2] ? max : Arr[i][2];

min = Arr[i][0] < Arr[i][1] ? Arr[i][0] : Arr[i][1];

min = min < Arr[i][2] ? min : Arr[i][2];

mid = Arr[i][0] + Arr[i][1] + Arr[i][2] - max - min;

if (max != 3 || mid != 2 || min != 1) //如果不符合规定

{

printf("该数独不正确\n");

return 0;

}

}

for (int i = 0; i < N;i++) //处理每一列 同理

{

max = Arr[0][i] > Arr[1][i] ? Arr[0][i] : Arr[1][i];

max = max > Arr[2][i] ? max : Arr[2][i];

min = Arr[0][i] < Arr[1][i] ? Arr[0][i] : Arr[1][i];

min = min < Arr[2][i] ? min : Arr[2][i];

mid = Arr[0][i] + Arr[1][i] + Arr[2][i] - max - min;

if (max != 3 || mid != 2 || min != 1)

{

printf("该数独不正确\n");

return 0;

}

}

printf("该数独正确\n");

return 0;

}

第六题

#include <stdio.h>

#define N 3

int main()

{

int Arr[N][N];

for (int i = 0; i < N;i++) //得到3x3的矩阵

{

for (int j = 0; j < N;j++)

{

scanf\_s("%d", &Arr[i][j]);

}

}

int temp;

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

{

for (int j = i + 1; j < N; j++) //如果数据在左上右下对角线右上角

{

temp = Arr[i][j];

Arr[i][j] = Arr[j][i]; //和对称位置交换

Arr[j][i] = temp;

}

}

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

{

for (int j = 0; j < N; j++)

{

printf("%4.d", Arr[i][j]); //输出

}

printf("\n");

}

return 0;

}

第七题

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include<string.h>

int main()

{

char Str1[200],Str2[100];

printf("请输入第一段文字\n"); //输入文字

scanf("%s", &Str1);

printf("请输入第二段文字\n");

scanf("%s", &Str2);

for (int i = strlen(Str1),j=0;i <= strlen(Str1) + strlen(Str2); i++,j++) //从str1的终止符循环，一直复制到str2末尾

{

Str1[i] = Str2[j];

}

printf("%s", Str1);

return 0;

}

第八题

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <string.h>

int main()

{

char Str1[666],Str2[666];

printf("请输入一个字符串\n"); //输入字符串

scanf("%s",&Str1);

int j = 0; //j表示str2待写入位

for (int i = 0; i <= strlen(Str1);i++) //循环字符串

{

if ((Str1[i] >= 48 && Str1[i] <= 57) || (Str1[i] >= 65 && Str1[i] <= 90) || (Str1[i] >= 97 && Str1[i] <= 122)) //判断符合要求

{

Str2[j] = Str1[i]; //将对应str1[i]写入str2[j],待写入位后移

j++;

}

}

Str2[j] = '\0'; //补充终止符

printf("%s", Str2);

return 0;

}

第九题

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <string.h>

int main()

{

char Input;

int index = 0;

printf("请输入一个字符串\n"); //输入字符串

do

{

Input = getchar();

if (Input == ' ') index++; //如果是空格，那么记录

} while (Input != '\n'); //循环到回车

printf("%d", index + 1); //单词数量是空格+1

return 0;

}

第十题

#include <stdio.h>

#define N 40 //筷子长度上限

int main()

{

int NumOFSize[N] = {0}; //用于统计长度对应的根数

int Size;

for (int i = 0; i < 7;i++)

{

scanf\_s("%d", &Size);

NumOFSize[Size]++; //长度对应的数量增加

}

int Count =0;

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

{

if (NumOFSize[i] % 2 != 0) //遍历所有长度

{

printf("漏掉的筷子长度为 %d\n", i); //不成对则输出

Count++; //不成对计数增加

}

}

if (Count != 1)

{

printf("但是有很多组不成对，找不到漏掉的筷子\n"); //有多个不成对则报错

}

return 0;

}

第十一题

#include <stdio.h>

#include <stdlib.h>

#define N 5 //历史记录长度

struct Calcul

{

bool Ready = false; //用于判定是否已经记录了结果

float a; //用于记录一次运算

float b;

char Operator;

float Result;

};

Calcul History[N]; //历史队列

void PrintMenu() //输出菜单

{

printf(

"------------菜单------------\n"

"操作：\n"

" 1. + 加\n"

" 2. - 减\n"

" 3. \* 乘\n"

" 4. / 除\n"

" 5. %% 取余\n"

" 6. 查看历史记录\n"

"请输入数字选项:"

);

}

void AddHistory(float a, float b, char Operator, float Result) //用于增加一个历史记录

{

for (int i = N-1; i >=1 ;i--) //把每一条历史向后挪一条，最后的丢弃

{

History[i].a = History[i - 1].a;

History[i].b = History[i - 1].b;

History[i].Operator = History[i - 1].Operator;

History[i].Ready = History[i - 1].Ready;

History[i].Result = History[i - 1].Result;

}

History[0].a = a; //腾出来的第一空位用来记录新的

History[0].b = b;

History[0].Operator = Operator;

History[0].Result = Result;

History[0].Ready = true;

};

void PrintHistory() //输出历史

{

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

{

if (History[i].Ready == true) //如果有历史记录，那么就输出

{

printf("%d: %.2f %c %.2f = %.2f \n", i + 1, History[i].a, History[i].Operator, History[i].b, History[i].Result);

}

else

{

break; //如果i条记录没有Ready，说明后面也为空

}

}

};

int main()

{

char chose;

char Operator;

float a, b, Result;

bool Done;

StratPos: //重新开始的标记

PrintMenu(); //输出菜单

chose = getchar();

Done = 0; //表明是否是一次成功的计算，是否可以记录进历史

if (chose >= '1' && chose <= '5') //如果是计算，那么获得操作数

{

printf("请输入两个操作数：\n");

scanf\_s("%f%f", &a, &b);

}

switch (chose) //按照菜单判断

{

case '1':

Result = a + b; //计算

Operator = '+'; //记录操作符

Done = 1; //表示成功计算，可以记录进历史

break;

case '2':

Result = a - b;

Operator = '-';

Done = 1;

break;

case '3':

Result = a \* b;

Operator = '\*';

Done = 1;

break;

case '4':

while (b == 0)

{

printf("除数不合法，请重新输入：\n");

scanf\_s("%f", &b);

}

Result = a / b;

Operator = '/';

Done = 1;

break;

case '5':

while (b == 0)

{

printf("取余数不合法，请重新输入：\n");

scanf\_s("%f", &b);

}

Result = (int)a % (int)b;

Operator = '%';

Done = 1;

break;

case '6': PrintHistory();break; //输出历史

default:

printf("输入选项不合法：\n"); //报错

break;

};

if (Done) //如果是成功的计算，那么输出并且进历史

{

printf("%.2f %c %.2f = %.2f",a, Operator, b, Result);

AddHistory(a, b, Operator, Result);

}

system("pause"); //最后暂停

getchar(); //和上面一句共用，因为pause之后，输入缓冲区里有一个按键，需要消除掉

system("cls"); //清楚屏幕，重新显示菜单

goto StratPos;

return 0;

}

拓展1

#include <stdio.h>

#define N 200

int main()

{

bool Arr[N + 1]; //判断index是不是一个素数

for (int i = 1; i <= N; i++)

{

Arr[i] = true; //假设所有的都是素数

}

for (int i = 2;i <= N;i++) //从二开始循环

{

if (Arr[i] == true) //如果那个数是素数（2肯定是)

{

for (int j = i + 1; j <= N; j++) //遍历它后面的数只要是倍数，都不是素数

{

if (j % i == 0)

{

Arr[j] = false; //不是素数进行标记

}

}

}

}

for (int i = 1; i <= N; i++) //输出

{

if (Arr[i] == true)

{

printf("%d ",i);

}

}

return 0;

}

拓展2

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <string.h>

int main()

{

char Str1[666] = {0}, Str2[666] = {0}; //注意，scanf只会补充一个\0.为了避免i=2n跳过溢出，需要全为0

printf("请输入一个字符串\n"); //输入字符串

scanf("%s", &Str1);

int i = 0, j = 0;

for (; Str1[i] != '\0'; i += 2, j += 1) //i一次增两个，j增一个，对应str1，str2赋值

{

Str2[j] = Str1[i];

}

printf("%s", Str2);

return 0;

}

拓展3

#include <stdio.h>

#define N 10

void OverFloor(int f1[], int f2[]) //用于将f2挪到f1，为下一层计算腾出空间

{

for (int i = 0; i <= N+4; i++)

{

f1[i] = f2[i];

f2[i] = 0;

}

}

int main()

{

int floor1[N + 5] = {0}, floor2[N + 5] = {0};

floor1[1] = 1; //先解决第一层

for (int i = 1; i <= N + 1;i++) //从第一层开始

{

for (int k = 1;floor1[k] != 0; k++) //先输出

{

printf("%4.d", floor1[k]);

}

printf("\n");

for (int j = 1; j <= i; j++) //计算下一层，每一层第j个元素，是对应下一层j、j+1的加数

{

floor2[j] += floor1[j];

floor2[j+1] += floor1[j];

} //迭代

OverFloor(floor1, floor2); //计算完成，清理f2，用于下一次计算

}

return 0;

}

拓展4

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

int main()

{

char PassWord[20];

scanf("%s", &PassWord); //输入

int PerNum[8];

for (int i = 0;i <= 7;i++) //每一位得到数字，+5 %10

{

PerNum[i] = PassWord[i] - '0';

PerNum[i] = (PerNum[i] + 5) % 10;

}

int Mov3PerNum[8]; //移动三位

for (int i = 5,j = 0; j <= 7;i++, j++)

{

Mov3PerNum[j] = PerNum[i % 8];

}

for (int i = 7; i >= 0; i--) //逆向输出

{

printf("%d", Mov3PerNum[i]);

}

return 0;

}

拓展5

#include <stdio.h>

#define N 5

int main()

{

bool away[N + 1] = {0};

int Say, ID, Now = 1, NumOfPlayer = N;

int count = 0;

for (int i = 1; i <= N; i++)

{

printf("\n现在从编号 %d 的人开始计数\n在场上还有\n",Now);

/\*for (int j = 1; j <= N; j++)

{

if (!away[j]) printf("%d ", j);

}\*/

printf("\n喊到几的人退出：\n");

Say = getchar() - '0';

getchar();

Say = Say % (NumOfPlayer + 1);

count = 0; ID = Now;

while (count != Say)

{

if (away[ID % (N + 1)] == false) count++;

ID = (ID +1) % (N + 1);

}

away[ID] = true;

NumOfPlayer--;

Now = ID;

while (away[Now] != false)

{

Now++;

}

if (NumOfPlayer == 1) break;

}

printf("最后一人，游戏结束");

return 0;

}

拓展6

#include <stdio.h>

#include <math.h>

#define N 10

void AbsBubbleRange(int Arr[], int Len) //冒泡排序，按照绝对值

{

int temp;

bool flag;

for (int i = 0; i <= Len - 2; i++)

{

flag = true;

for (int j = 0; j <= Len - 2 - i;j++)

{

if (abs(Arr[j + 1]) < abs(Arr[j])) //绝对值

{

temp = Arr[j + 1];

Arr[j + 1] = Arr[j];

Arr[j] = temp;

flag = false;

}

}

if (flag == true)break;

}

}

int main()

{

int a[N], b[N];

for (int i = 0; i < N; i++) //输入ab

{

scanf\_s("%d", &a[i]);

b[i] = a[i];

}

AbsBubbleRange(b, N); //将b排序

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

{

printf("%d ", a[i]);

}

printf("\n");

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

{

printf("%d ", b[i]);

}

return 0;

}