```
// 基礎プロ I 100 本ノック 中級編

//------------------------------
/*
No. 40 even or odd
整数値を入力させ、その値が偶数ならば even、奇数ならば odd と表示す
るプログラムを作成せよ。
【実行例、下線部は入力例】
$ ./knock40
input number: 6
6 is even.
$ ./knock40
input number: 7
7 is odd.
$
偶数＝2 で割り切れる＝2 で割った余りが 0


*/

#include <stdio.h>

int main(void)
{
 int input_num = 0;

 printf("input number: ");
 scanf("%d", &input_num);

 if (input_num % 2 == 0)
 {
  printf("even");
 }
 else
 {
  printf("odd");
 }
 return 0;
}
```

```
//------------------------------
/*
No. 41 1 桁の自然数?
整数値を入力させ、その値が一桁の自然数かそうでないか判定するプログ
ラムを作成せよ。
【実行例、下線部は入力例】
$ ./knock41
input number: 6
6 is a single figure.
$ ./knock41
input number: 10
10 is not a single figure.
$ ./knock41
input number: -3
-3 is not a single figure.
$ ./knock41
input number: 0
0 is not a single figure.
$
一桁の自然数＝0 より大きく、かつ、9 以下の整数、として判定すればよい。
*/
#include <stdio.h>

int main(void)
{
 int input_num = 0;
 printf("input number: ");
 scanf("%d", &input_num);

 if (input_num > 0 && input_num < 9)
 {
  printf("%d  is a single figure.", input_num);
 }
 else
 {
  printf("%d  is not a single figure.", input_num);
 }
 return 0;
}
/*
自然数とは「正の整数」を意味する言葉ですが、0 より大きな整数、
つまり「0 を含まない正の整数」である
*/
//------------------------------
```

```
/*
No. 42 小さい順?
整数値を 3 つ入力させ、それらの値が小さい順（等しくてもよい）に並ん
でいるか判定し、小さい順に並んでいる場合は OK、そうなっていない場合
は NG と表示するプログラムを作成せよ。

*/
// #1


#include <stdio.h>

int main(void)
{
 int input_num1 = 0;
 int input_num2 = 0;
 int input_num3 = 0;

 printf("input number1: ");
 scanf("%d", &input_num1);

 printf("input number2: ");
 scanf("%d", &input_num2);

 printf("input number3: ");
 scanf("%d", &input_num3);


 if(input_num1 <= input_num2 && input_num1 <= input_num3)
 {
  printf("OK");
 }
 else{
  printf("NG");
 }
 return 0;
}




// #2
```

```c
#include <stdio.h>

int main(void)
{
 int input_num[3] = {0};
 int flg = 0;


 for (int i = 0; i < 3; i++)
 {
  printf("input number%d: ", i + 1);
  scanf("%d", &input_num[i]);
 }

 if(input_num[0] <= input_num[1] && input_num[0] <= input_num[2])
 {
  flg = 1;
 }

 switch (flg)
 {
 case 1:
  printf("OK");
  break;

 default:
  printf("NG");
  break;
 }
 return 0;
}
```

```
//--------------------------------
/*
No. 43 2 次方程式の解の判別
2 次方程式 ax^2 + bx + c = 0 （x^2 は x の 2 乗の意味）の係数 a, b, c
を入力し、2 次方程式の解が 2 つの実数解か、重解か、2 つの虚数解かを判
別するプログラムを作成せよ。
【実行例、下線部は入力例】
$ ./knock43
input a: 4
input b: -3
input c: 1
2 つの虚数解
$ ./knock43
input a: 1
input b: 2
input c: 1
重解
$ ./knock43
input a: 3
input b: 2
input c: -4
2 つの実数解
$


*/

#include <stdio.h>

int main(void)
{
 int a = 0;
 int b = 0;
 int c = 0;

 int d = 0; // 判別式 : D = b2 -4ac

 /*
 2 次方程式の実数解の個数
 D > 0 ⇔ 異なる実数解 2 つ
 D = 0 ⇔ 実数解を 1 つ（重解）
 D < 0 ⇔ 実数解なし
 */
 printf("input a: ");
```

```
 scanf("%d", &a);

 printf("input b: ");
 scanf("%d", &b);

 printf("input c: ");
 scanf("%d", &c);

 d = (b * b) - (4 * a * c);

 if (d > 0)
 {
  printf("2つの実数解");
 }

 if (d == 0)
 {
  printf("重解");
 }

 if (d < 0)
 {
  printf("２つの虚数解");
 }

 return 0;
}
```

```
//--------------------------------
/*
No. 44 通貨換算
換算したい金額（円単位）と 1 ドル何円かを整数値で入力すると、入力し
た金額が何ドル何セントか計算して表示するプログラムを作成せよ。1 セ
ントは 1/100 ドルである。結果は整数値でよい（1 セント未満の端数切り
捨て）。


【実行例、下線部は入力例】
$ ./knock44
何円? 10000
1 ドルは何円? 120
10000 円は 83 ドル 33 セント
$ ./knock44
何円? 15000
1 ドルは何円? 125
15000 円は 120 ドル 0 セント


まず何ドルになるかを整数値で計算する。次に、1 ドルに満たない金額をセントに換算す
る。浮動小数点型の変数を使ってもよいが、使わない方法を考えてみよう。
$


*/

#include <stdio.h>

int main(void)
{
 int convert_money_en = 0;
 int one_dollar_en = 0;

 int dollar = 0;
 int cent = 0;

 printf("何円？ ");
 scanf("%d", &convert_money_en);

 printf("１ドルは何円？ ");
 scanf("%d", &one_dollar_en);

 dollar = convert_money_en / one_dollar_en;
```

```
 cent = convert_money_en % one_dollar_en * 100 / one_dollar_en;

 printf("%d 円は%d ドル%d セント\n", convert_money_en, dollar, cent);

 return 0;
}
```

```
//--------------------------------
/*
No. 45 タクシー料金
初乗り料金が 1700m まで 610 円、それ以降は 313m ごとに 80 円のタク
シーがある。このタクシーに乗った距離を m 単位で入力し、料金を計算す
るプログラムを作成せよ。
【実行例、下線部は入力例】
$ ./knock45
距離 10000
金額 2770
$ ./knock45
距離 2013
金額 690
$ ./knock45
距離 2014
金額 770
$

313m ごとの区間は 1m でも進んでしまったら 80 円かかることに注意。

*/

#include <stdio.h>

int main(void)
{
 int distance;
 int rest_distance;
 int money = 610;

 printf("%s", "距離 ");
 scanf("%d", &distance);



 if (distance <= 1700)
 {
  printf("%s%d", "金額 ", money);
 }



 if (distance > 1700)
 {
```

```
  rest_distance = distance - 1700;


  int roop_cnt = rest_distance /313;
   if(rest_distance % 313 != 0)
   {
    roop_cnt +=1;
   }



   for (int i = 0;i < roop_cnt; i++)
   {
    money += 80;
   }

   printf("%s%d", "金額 ", money);


 }

 return 0;
}

/*
料金の計算については、2 つの if 文で構成しました。
一つ目の if 文では、1700m までの料金の表示を行い、
二つ目の if 文では、1700m とその後の料金の計算を行いました。
1700m 後の 310m が何回あるかを for 文のループ回数に置き換え計算しました。
距離の端数の計算は、ループ回数に 1 を加算することで計算しました。



*/
```

```
//--------------------------------
/*
No. 46 入場料
神山美術館の入場料金は、一人 600 円であるが、5 人以上のグループなら
一人 550 円、20 人以上の団体なら一人 500 円である。人数を入力し、入
場料の合計を計算するプログラムを作成せよ。
【実行例、下線部は入力例】
$ ./knock46
人数 3
料金 1800
$ ./knock46
人数 7
料金 3850
$ ./knock46
人数 22
料金 11000
$


*/

#include <stdio.h>

int main(void)
{
 int person;
 int charge;


 printf("%s", "人数 ");
 scanf("%d", &person);




 if(person > 0 && person <= 4)
 {
  printf("%s%d\n", "料金 ",person * 600);
 }

 if(person >= 5 && person < 20)
 {
  printf("%s%d\n", "料金 ",person * 550);
 }
```

```
 if(person >= 20)
 {
  printf("%s%d\n", "料金 ",person * 500);
 }



 return 0;
}
```

```
//--------------------------------
/*
No. 47 値の入れ替え
異なる整数値を 2 つ入力し、それぞれ別の変数に格納する。そして、それ
らの変数の値を入れ替えた後、それぞれの変数の値を表示するプログラム
を作成せよ。単に順序を変えて表示するだけではダメ。必ず変数の値を入
れ替えること。下の実行例の場合、まず変数 a に 2、b に 5 が入力された状
態から、a の値が 5、b の値が 2 になるように入れ替える。
【実行例、下線部は入力例】
$ ./knock47
input a: 2
input b: 5
a = 5, b = 2
$


*/

#include <stdio.h>

int main(void)
{
 int a,b,x;



 printf("%s", "input a: ");
 scanf("%d", &a);


 printf("%s", "input b: ");
 scanf("%d", &b);

 x = a;
 a = b;
 b = x;
 printf("%s%d%s%d\n", "a = ",a,", b = ",b);



 return 0;
}
```

```
//--------------------------------

/*
No. 48 繰り返し計算
最初に 2 以上の整数値を入力し、次の規則で計算し、計算回数と計算結果
を表示し、計算結果が 1 になるまで繰り返すプログラムを作成せよ。
規則：ある値が偶数ならその値を 1/2 にする。奇数ならその値を 3 倍して
1 を足す。
【実行例、下線部は入力例】
$ ./knock48
input number: 3
1: 10
2: 5
3: 16
4: 8
5: 4
6: 2
7: 1
$


*/

#include <stdio.h>

int main(void)
{
 int a, b, x;

 int i = 0;

 printf("%s", "input number: ");
 scanf("%d", &a);

 if (a % 2 == 1)
 {
  b = a * 3 + 1;
  x = b;
 }

 if (a % 2 == 0)
 {
  b = a / 2;
  x = b;
```

```
 }

 for (i = 1; x != 1; i++)
 {
  printf("%d %d\n", i, x);
  a = x;

  if (a % 2 == 0)
  {
   x = a / 2;
  }

  if (a % 2 == 1)
  {
   x = a * 3 + 1;
  }
 }

 printf("%d %d\n", i, x);

 return 0;
}
```

```
//--------------------------------
/*
No. 49 九九
九九の表を、2 重の繰り返しを使って表示するプログラムを作成せよ。2 重
の繰り返しを使わず単に表示するだけではダメ。値の間はタブ(\t)を使っ
て間をあけること。
【実行例】
$ ./knock49
1 2 3 4 5 6 7 8
9
2 4 6 8 10 12 14 16
18
3 6 9 12 15 18 21 24
27
4 8 12 16 20 24 28 32
36
5 10 15 20 25 30 35 40
45
6 12 18 24 30 36 42 48
54
7 14 21 28 35 42 49 56
63
8 16 24 32 40 48 56 64
72
9 18 27 36 45 54 63 72
81
$

*/

//# 1 実行例と同じ結果

#include <stdio.h>

int main(void)
{
 int i,j;

 for (i = 1; i < 10; i++)
 {
  for (j = 1; j < 10; j++)
  {
   printf("%d\t", i * j);
   if(j % 8 == 0)
```

```
    {
     printf("\n");
    }
   }

   printf("\n");
  }

  return 0;
}


//#2 改
#include <stdio.h>

int main(void)
{
  int i,j;

  for (i = 1; i < 10; i++)
  {
   for (j = 1; j < 10; j++)
   {
    printf("%d\t", i * j);

   }

   printf("\n");
  }

  return 0;
}









//--------------------------------
```

```
/*
No. 50 foobar
1 から 100 までの値を繰り返しで表示するが、3 の倍数の時は foo、5 の倍
数の時は bar と数字の代わりに表示するプログラムを作成せよ。なお、3
と 5 の両方の倍数の時は foobar と表示される。
【実行例】
$ ./knock50
1
2
foo
4
bar
foo
（途中省略）
97
98
foo
bar
$


*/

#include <stdio.h>

int main(void)
{


 for (int i = 1; i <= 100; i++)
 {
  if (i % 3 == 0)
  {
   if (i % 5 == 0)
   {
    printf("foobar\n");
   }
   else
   {
    printf("foo\n");
   }
  }
  else if (i % 5 == 0)
  {
```

```
    printf("bar\n");
   }
   else
   {
    printf("%d\n", i);
   }
  }

  return 0;
}
```

```
//------------------------------
/*
```

```
No. 51 お支払い
指定した金額を 100 円玉と 10 円玉と 1 円玉だけで、できるだけ少ない枚
数で支払いたい。金額を入力するとそれぞれの枚数を計算して表示するプ
ログラムを作成せよ。
【実行例、下線部は入力例】
$ ./knock51
input money: 12345
100 円玉 123 枚, 10 円玉 4 枚, 1 円玉 5 枚
$
問題文は「できるだけ少ない」となっているが、金額の大きな硬貨を優先して使うように
計算すれば自然とそうなる。


*/

#include <stdio.h>

int main(void)
{
 int input_money;
 int en_100, en_10, en_1;

 printf("input money: ");
 scanf("%d", &input_money);

 en_100 = input_money / 100;

 en_10 = input_money % 100 / 10;

 en_1 = input_money % 10;

 printf("100 円玉 %d 枚、10 円玉 %d 枚、1 円玉 %d 枚\n", en_100, en_10, en_1);
 return 0;
}
 /*
 printf("%.2f\n", 12345 / 100.00);
 printf("%d", 12345 % 100 / 10);printf("  (%.2f)\n", 12345 % 100 / 10);
 printf("%d", 12345 % 10);printf("   (%.2f)\n", 12345 % 10);
 //小数桁を整数にして計算する

 */
//-------------------------------

/*
```

```
No. 52 閏年
西暦を入力したらその年が閏（うるう）年かどうかを判定するプログラム
を作成せよ。なお、4 で割り切れる年のうち、100 で割り切れないか、
400 で割り切れる年は閏年である。
【実行例、下線部は入力例】
$ ./knock52
input year: 2015
2015 年は閏年でない
$ ./knock52
input year: 2016
2016 年は閏年である
$ ./knock52
input year: 2100
2100 年は閏年でない
$ ./knock52
input year: 2000
2000 年は閏年である
$


*/

#include <stdio.h>

int main(void)
{
 int input_year;


 printf("input year: ");
 scanf("%d", &input_year);

 if(input_year % 4 == 0)
 {
  if(input_year % 100 != 0 || input_year % 400 == 0){

   printf("%d は閏年である\n",input_year);
  }else
 {
  printf("%d は閏年でない\n",input_year);

 }
 }else
 {
```

```
   printf("%d は閏年でない\n",input_year);

  }
  return 0;
}


/*
ルール①：うるう年は西暦年号が 4 で割り切れる年
ルール②：西暦年号が 100 で割り切れて 400 で割り切れない年は平年


うるう年の意味
うるう年（閏年）とは、2 月 29 日が存在する年のことをいいます。「閏（うるう）」とは、
暦こよみの上での日数や月数が平年より多いことを指し、この日を「閏日（うるうび）」、
閏日がある月を「閏月（うるうづき）」、閏日がある年を「閏年（うるうどし、じゅんねん）」
といいます。
このうるう年は、約 4 年に一度訪れます。平年は 365 日ですが、うるう年には 366 日になると
いうことです。
なぜ、うるう年が設定されたのかというと、実際の季節と暦がずれてしまうためです。
平年を 365 日とする太陽暦で、地球の平均回帰年（太陽が黄道上の分点と至点から出て再び各点
に
戻ってくるまでの周期のこと）は約 365.242199 日です。ですので、ずっと 365 日の暦にして
しまうと、徐々に季節と暦がずれてしまいます。そのため、ほぼ 4 年に一度、2 月に 1 日を足して
調節をしているのです。
*/









//--------------------------------
```

```
/*
No. 53 素因数分解
自然数の入力値を素因数分解して結果を表示するプログラムを作成せよ。
【実行例、下線部は入力例】
$ ./knock53
input number: 840
2 2 2 3 5 7
$

*/

#include <stdio.h>

int main(void)
{
 int input_number;
 int prime_number; // 素数
 int quotient;   // 商


 printf("input number: ");
 scanf("%d", &input_number);


 quotient = input_number;


 for (prime_number = 2; quotient != 1; prime_number++)
 {

  for (int i = 0;quotient % prime_number == 0 ; i++)
  {

   quotient = quotient / prime_number;
   printf("%d ", prime_number);

  }

 }
 return 0;
}

/*
for 文のネストを使って問題を解きました。
```

１つ目の for 文では、素数（prime_number)を設定して、
２つ目の for 文で　入力値を素数で割り,商の余りを計算をして、
商の余りが出たときに１つ目の for 文で設定した素数の値が一つ上がり、
商の余りが１になるまで繰り返すという仕組みになっているよう考えました。
*/

//--------------------------------

```
/*
No. 54 最大最小
まずデータの個数を入力させ、次にデータの個数だけ整数値を入力させる。
この入力データの中で最大値と最小値を求め表示するプログラムを作成せ
よ。データの個数は 100 個までとする。なお、データの個数とデータは
ファイルからリダイレクトで入力させればよいので、入力のためのメッ
セージは不要である（実行例を参照すること）。
【実行例、データファイルは下のリンクから取得せよ】
$ ./knock54 < small.data
最小値 = 128, 最大値 = 962
$ ./knock54 < middle.data
最小値 = 20, 最大値 = 988
$ ./knock54 < large.data
最小値 = 5, 最大値 = 996
$
リダイレクトでデータを入力するプログラム例はこちら redirect.c
入力データはこちら（右クリックで「リンク先をダウンロード」。ブラウザの設定によっ
てはファイルの拡張子に.txt が追加されるが、その場合は適宜変更すること。「別名でダ
ウンロード」を選んで、プログラムを置いているディレクトリに保存するとよい。）
small.data
middle.data
large.data
*/

/****
   redirectinput.c
   リダイレクトでデータを入力させる
   データの形式は、
   1 行目にデータの個数、
   2 行目以降は各データの値
   使用方法：
   redirectinput < datafile
*****/

#include <stdio.h>


typedef struct read_f
{
 char file[1];
} read_f;

int main()
{
```

```
FILE *fp;
int numData;   // データの個数
int data[100]; // データを格納する配列、100 個分
int i;         // カウンタ
int a, b;

a = 0;    // 最小値初期値
b = 9999; // 最大値初期値



read_f rf[1];

scanf("%s", &rf[0].file); //=====start： file 名.data 入力======

fp = freopen(rf[0].file, "r", stdin);

scanf("%d", &numData); // データの個数を読み込む

fclose(fp);



//データの内容読み込み s
freopen(rf[0].file, "r", stdin);

for (i = 0; i < numData; i++)
{
 scanf(" %d", &data[i]);



//データ比較（最小値、最大値）
  if (a < data[i])
  {
   a = data[i];
  }
  if (b > data[i])
  {
   b = data[i];
  }
 }
```

```
 // 表示（データ個数、最大値・最小値）
 printf("numData = %d\n", numData);
 printf("最小値 = %d  最大値 = %d\n", b, a);
 fclose(fp);



 return 0;
}



/*
主なプログラムの構成
１，データの個数読み込み　部分
２，データの内容読み込み　部分
３，データ比較（最小値、最大値）部分
４，表示（データ個数、最大値・最小値）部分

難しかったところ
・リダイレクト入力の理解
・scanf 関数の扱い（数値代入のための変数宣言、リダイレクト入力での使用）
・ファイルの入出力処理
・構成（一連の動作にまとめる）
・問題の理解

何度も試行錯誤を繰り返し、一応の完成に至りました。
問題内容の理解にもつまずき、進んで戻っての繰り返しでしたが
学びの多い問題でした。

*/

/*
   fgets 関数：ファイルを 1 行ずつ読み取る。

   fgets 関数の第 1 引数には 1 行分の文字列を格納する配列を
   、第 2 引数には 1 行の最大文字数を入力し
   、第 3 引数に FILE 型構造体の実体のアドレスを入力します。
   戻り値には文字列のポインタを返し、失敗すると NULL を返します。


   stdin
```

```
標準入力とは何ですか？
標準入力・出力とはなんですか？
標準入力はプログラムに値を渡す入力元のこと、
標準出力はプログラムから出力される値の出力先のことをいいます。
この仕組みを使うと、キーボードやファイルなど外部から値（データ）を
プログラムに与えることができます


*/









//--------------------------------
```

```
/*
No. 55 夢想花 again
「とんで」を 9 回「まわって」を 3 回繰り返した後「まわる」と表示して
改行する、を 3 回繰り返すプログラムを作成せよ。「とんで」「まわっ
て」と 3 行文の繰り返しは必ず繰り返し構文を使うこと。
【実行例】
$ ./knock55
とんでとんでとんでとんでとんでとんでとんでとんでとんでまわってまわってまわってまわる
とんでとんでとんでとんでとんでとんでとんでとんでとんでまわってまわってまわってまわる
とんでとんでとんでとんでとんでとんでとんでとんでとんでまわってまわってまわってまわる
$s

*/

#include <stdio.h>

int main()
{

  for(int i = 0; i<3; i++)
  {

   for(int j = 0; j < 9; j++)
   {
     printf("とんで");
   }

   for(int k = 0; k < 3; k++)
   {
     printf("まわって");
   }
   printf("まわる\n");

  }

  return 0;
}




//--------------------------------
```

```
/*
No. 56 2 進数変換
0～65535 の整数値を入力させ、入力値を 16 桁の 2 進数に変換して表示す
るプログラムを作成せよ。
【実行例、下線部は入力例】
$ ./knock56
input number: 127
0000000001111111
$ ./knock56
input number: 10000
0010011100010000
$ ./knock56
input number: 65535
1111111111111111
$
ヒント：16 桁分の 2 進数を記憶する配列を用意する。2 で割った余り(0 か 1)を最初の桁
の値とし、2 で割った値を新たな値とし、さらに 2 で割った余りを次の桁の値とする。こ
れを繰り返していけば 1 の桁から順に値が求まるので、表示するときは逆順に表示すれば
よい。


*/

#include <stdio.h>

int main()
{

  int array_01[16];
  int input_number;
  int i = 0;

  printf("input number:");
  scanf("%d", &input_number);

  for (i; i <= 15; i++)
  {
    if (input_number % 2 != 0)
    {
      array_01[i] = 1;
    }
    else
    {
      array_01[i] = 0;
```

```
    }
    input_number = input_number / 2;
  }

  for (i=15; i >= 0; i--)
  {
    printf("%d", array_01[i]);
  }



  return 0;
}
```

```
/*
ヒントの通り
int 配列 16 桁分を定義して、input number(input_number)を
2 で割り、割り切れなければ、配列に 1 を、
割り切れれば（余り 0)、配列に 0 を、代入しました。

数値の入力部分の他、
配列への代入、そのための条件、入力数値の商の部分と
代入した数値の表示の部分との
上下 2 つの for 文で構成しました。

*/

/*
配列  int array_01[16] ={3,3,3,........};
// ... の代入していないところは  0 が入る（？？？？）
*/

/*
配列のはみ出し(?? outofbounds)
// java とは異なる。
???? C 言語、配列の桁越え（outofbounds)
？？メモリ内処理の関係上？？？？
詳しくは不明、必要なら検索
*/

//------------------------------
```

```
/*
No. 57 テスト集計
```
まず受験者数を入力させ、次に受験者数ごとに英語、数学、国語の点数を
スペースで区切って入力させる（具体的な入力方法は下記の scanf の使い
方の説明、および入力データの中身を見よ）。入力が終了したら、英語、
数学、国語の平均点、および各受験生の合計点を計算して表示するプログ
ラムを作成せよ。受験者数は 100 人までとする。なお、データの個数と
データはファイルからリダイレクトで入力させればよいので、入力のため
のメッセージは不要である（実行例を参照すること）。
【実行例、データファイルは下のリンクから取得せよ】
```
$ ./knock57 < examSmall.data
平均点 英語:46, 数学:51, 国語:55
個人合計点
0: 141
1: 114
（途中省略）
8: 96
9: 188
$ ./knock57 < examMiddle.data
平均点 英語:55, 数学:53, 国語:54
個人合計点
0: 136
1: 64
（途中省略）
48: 265
49: 167
$ ./knock57 < examLarge.data
平均点 英語:52, 数学:51, 国語:51
個人合計点
0: 151
1: 241
（途中省略）
98: 107
99: 178
$
```
複数の値をスペースで区切って一度に入力させるには、次のように scnaf 関数の書式指定
文字列を使う：
```
scanf("%d %d %d", &eng, &math, &lang);
```
入力データはこちら（右クリックで「リンク先をダウンロード」）
examSmall.data

examMiddle.data
examLarge.data
*/

```c
#include <stdio.h>

typedef struct read_f
{
 char file[1];
} read_f;

int main()
{
 FILE *fp;
 int numexam;              // 受験者数
 int data[3][100] = {0}; // データを格納する配列、100 個分
 int sum_data_avg[3] = {0};
 int average[3] = {0};
 int sum_one_person[100] = {0};
 int i; // カウンタ



 read_f rf[1];
 scanf("%s", &rf[0].file); //=====start： file 名.data 入力======



 fp = freopen(rf[0].file, "r", stdin);
 scanf("%d", &numexam);         // 受験者数を読み込む
 for (i = 0; i < numexam; i++) // データの内容読み込み
 {
  scanf("%d %d %d", &data[0][i], &data[1][i], &data[2][i]);




  sum_data_avg[0] += data[0][i]; // 各平均値計算のための各合計値計算
  sum_data_avg[1] += data[1][i];
  sum_data_avg[2] += data[2][i];

  sum_one_person[i] = data[0][i] + data[1][i] + data[2][i];
   //受験生一人当たりの合計点
```

```
 //表示
  //printf("%d  %d %d %d\n",i+1, data[0][i], data[1][i], data[2][i]);
 }
 fclose(fp);


 for (int j = 0; j < 3; j++)
 {

  average[j] = sum_data_avg[j] / numexam;
 }
 printf("平均点  英語:%d,  数学:%d,  国語:%d\n", average[0], average[1],
average[2]);


 printf("個人合計点\n");
 for (int k = 0; k < numexam; k++)
 {

  printf("%d: %d\n", k+1, sum_one_person[k]);
 }



 return 0;
}

/*
プログラムの主な構成
1，フィールド（変数定義）
2，ファイル名、入力部分
3，データの読み込み部分
4，計算部分（平均点、合計点）
5，表示部分


No.54 で作成したコードを再利用して作成しました。
ファイルデータの１行目（受験者数）とそれ以降（点数データ）の
読み込み部分のコードを考えるのが時間がかかりました。
```

```
*/
//FILE *fp;  (ポインタ変数)

//--------------------------------

/*
No. 58 棒グラフ
0 以上の整数値を 5 つ入力させ、それぞれの値に対して、次の形式でグラフを描くプログラムを作成せよ。
形式：左端に値を表示し、適切に空白を空けて":"を書く（:で揃えるためにタブ\t を使うとよい）。その後ろに値の数だけ*を描くが、5 個おきに空白を１つ入れる。具体例は下記の実行例を参照すること。
【実行例、下線部は入力例】
$ ./knock58
input data[0]: 7
input data[1]: 10
input data[2]: 14
input data[3]: 15
input data[4]: 21
7 :***** **
10 :***** *****
14 :***** ***** ****
15 :***** ***** *****
21 :***** ***** ***** ***** *
$
ヒント：入力値は配列に格納する。グラフを描くためには、値のループと、*を描くループの二重ループとなる。


*/

#include <stdio.h>

int main()
{
 int input_data[5] = {0};
 int i, j, k;

 for (i = 0; i < 5; i++)
 {
  printf("input data[%d]: ", i);
  scanf("%d", &input_data[i]);
 }
```

```
 for (j = 0; j < 5; j++)
 {
  printf("%d\t:", input_data[j]);

  for (k = 0; k < input_data[j]; k++)
  {
   if (k % 5 == 0)
   {
    printf(" ");
   }
   printf("*");
  }
  printf("\n");
 }

 return 0;
}
```

```
//------------------------------


/*
No. 59 行列の和
3x3 行列の和を求めて表示するプログラムを作成せよ。行列の値は 2 次元
配列で表現し、繰り返しを使って計算すること。
3x3 行列とは縦 3 つ、横 3 つの 9 つの要素(値)をひとまとめにして扱うも
のである。2 つの 3x3 行列の和は次式のように、それぞれ同じ位置にある
値を足したものとして計算できる。
例えば a12 という要素は、1 行目 2 列目の要素という意味である。それぞ
れ同じ位置にある要素を足せばよい。
なお、入力値は 1 行ずつ 3 つの値をスペースで区切って入力するようにす
るとよい。このためには、scanf("%d %d %d", &a[0][0], &a[0][1],
&a[0][2]);のように書く(No. 57 参照)。
【実行例、下線部は入力例】
$ ./knock59
1 つめの行列
1 2 3
4 5 6
7 8 9
1 つめの行列
2 3 4
5 6 7
8 9 1
和
3 5 7
9 11 13
15 17 10
$
C 言語の配列の添字は 0 から始まることに注意。上記の実行例で、和の行列の表示は各列
の値を揃えるためにタブ(\t)を使っている。

*/
//#1  a[3][3], b[3][3], c[3][3]  パターン

#include <stdio.h>

int main()
{

 int a[3][3], b[3][3], c[3][3];
```

```
 int i;
 int y, x;
 int count = 1;

printf("1つ目の行列\n");
for(i = 0; i<3; i++)
{

scanf("%d %d %d", &a[i][0], &a[i][1], &a[i][2]);
}

 printf("2つ目の行列\n");
for(i = 0; i<3; i++)
{
scanf("%d %d %d", &b[i][0], &b[i][1], &b[i][2]);
}


 printf("和\n");
 for (y = 0; y < 3; y++)
 {
  for (x = 0; x < 3; x++)
  {
   c[y][x] = a[y][x] + b[y][x];

   printf("%d\t", c[y][x]);
  }
  printf("\n");
 }

 return 0;
}




//#2  a[9][3]  パターン

#include <stdio.h>

int main()
{
```

```c
 int a[9][3];

 int i;
 int y,x;
 char count[4] = {"1or2"};




 for (i = 0; i < 6; i++)
 {
  if (i == 0 || i == 3)
  {
   printf("%c つ目の行列\n", count[i]);
  }

  scanf("%d %d %d", &a[i][0], &a[i][1], &a[i][2]);
 }




 printf("和\n");
 for (y = 0; y < 3; y++)
 {
  for (x = 0; x < 3; x++)
  {
   a[y+6][x] = a[y][x] + a[y+3][x];

   printf("%d\t", a[y+6][x]);
  }
  printf("\n");
 }


 return 0;
}

/*
配列が[3][3] と [9][3]の２パターンのコードを
作成しました。
基本的には同じつくりですが[9][3]は少し悩みました。

*/
```