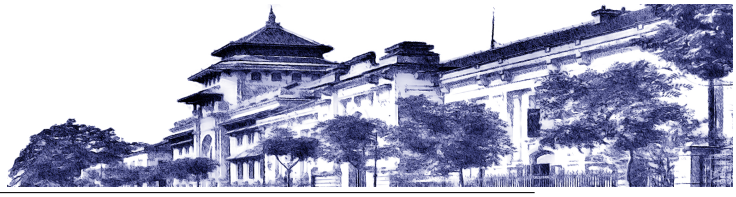
ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
Khoa Vật lý, Bộ môn Tin học Vật lý
P. 308D-T1, 334 Nguyễn Trãi, Thanh Xuân, Hà Nội



Phần 8

Nội dung thực hành

– Cấu trúc: *Định nghĩa, khai báo, truy cập các thành phần biến cấu trúc (struct)*
– File: *Đóng/mở file, ghi/đọc dữ liệu trên file*

Ví dụ

1. Tạo cấu trúc lưu giữ thông tin cơ bản của sinh viên, rồi đọc vào và in ra thông tin đó

```c
#include <stdio.h>
// Cach 1: (nen dung cach nay)                //     // Cach 2:
typedef struct {                              //     struct sinhvien {
    int ns;       // nam sinh                 //         int ns;
    char ht[100]; // ho ten                   //         char ht[100];
    float dtb;    // diem trung binh          //         float dtb;
} sinhvien;                                   //     };

int main() {
    sinhvien a;                               //     struct sinhvien a;
    puts("Nhap vao ho va ten:");    gets(a.ht);
    puts("Nhap vao nam sinh: ");    scanf("%d", &a.ns);
    puts("Nhap vao diem tb:  ");    scanf("%f", &a.dtb);
    // In ra thong tin
    printf("Diem trung binh cua sv %s (sinh nam %d) la: %f \n", a.ht, a.ns, a.dtb);
    return 0;
}
```

2. Cấp phát bộ nhớ động cho con trỏ cấu trúc

```c
#include <stdio.h>
#include <stdlib.h>

typedef struct {
    int ns;       // nam sinh
    char ht[100]; // ho ten
    float dtb;    // diem trung binh
} sinhvien;

int main() {
    sinhvien *A;
    int i, n;
    printf("Nhap vao so sv: "); scanf("%d%*c", &n);
    // Cap phat bo nho cho mang A co n phan tu co kieu du lieu 'sinhvien'
    A = (sinhvien*)malloc(n * sizeof(sinhvien));
    // Nhap vao thong tin sinh vien
    printf("Nhap thong tin cua %d sv:\n", n);
    for (i = 0; i < n; i++) {
        printf("Ho va ten:  "); scanf("%25[^\n]", (A+i)->ht); // hoac: gets((A+i)->ht);
        printf("Nam sinh:   "); scanf("%d", &(A+i)->ns);
        printf("Diem tb:    "); scanf("%f%*c", &(A+i)->dtb);
    }
    // In ra thong tin da nhap
    printf("----------Thong tin sinh vien---------\n");
    printf("%-25s%-9s%s\n", "Ho va ten", "Nam sinh", "Diem");
    for (i = 0; i < n; i++)
        printf("%-25s%-9d%.2f\n", (A+i)->ht, (A+i)->ns, (A+i)->dtb);

    free(A);
    return 0;
}
```

3. Đọc vào dữ liệu sinh viên từ file, rồi in ra màn hình thông tin đó

```c
#include <stdio.h>
int main() {
    int i, n;
    int ns[20];          // nam sinh
    char ht[20][100];    // ho ten
    float dtb[20];       // diem trung binh
    // Doc vao thong tin sinh vien tu file "data_input.txt"
    FILE *f = fopen("data_input.txt", "r");
    fscanf(f, "%d%*c", &n);              // Doc vao so sinh vien
    for (i = 0; i < n; i++) {
        fscanf(f, "%25[^\n]", ht[i]);    // Doc vao ho ten
        fscanf(f, "%d", &ns[i]);         // Doc vao nam sinh
        fscanf(f, "%f%*c", &dtb[i]);     // Doc vao diem trung binh
    }
    fclose(f); // Ket thuc viec doc thong tin sinh vien bang cach dong file
    // In ra thong tin da doc
    printf("----------Thong tin sinh vien---------\n");
    printf("%-25s%-9s%s\n", "Ho va ten", "Nam sinh", "Diem");
    for (i = 0; i < n; i++)
        printf("%-25s%-9d%.2f\n", ht[i], ns[i], dtb[i]);
    return 0;
}
```

File *data_input.txt*:

```
3
Cong Phuong Cao           1996     9.99
Nguyen Thi Kieu Trang     1997     9.97
La Thi Ngoc Mai           1996     9.98
```

Màn hình:

```
----------Thong tin sinh vien---------
Ho va ten                 Nam sinh Diem
Cong Phuong Cao           1996     9.99
Nguyen Thi Kieu Trang     1997     9.97
La Thi Ngoc Mai           1996     9.98
```

4. Nhập thông tin sinh viên từ bàn phím, rồi ghi ra dữ liệu đó vào file

```c
#include <stdio.h>
int main() {
    int i, n;
    int ns[20];          // nam sinh
    char ht[20][100];    // ho ten
    float dtb[20];       // diem trung binh
    // Nhap vao thong tin sinh vien
    printf("Nhap vao so sv: "); scanf("%d%*c", &n);
    printf("Nhap thong tin cua %d sv:\n", n);
    for (i = 0; i < n; i++) {
        printf("Ho va ten:  "); scanf("%25[^\n]", ht[i]); // hoac: gets(ht[i]);
        printf("Nam sinh:   "); scanf("%d", &ns[i]);
        printf("Diem tb:    "); scanf("%f%*c", &dtb[i]);
    }
    // Ghi ra thong tin sinh vien vao file "data_output.txt"
    FILE *f = fopen("data_output.txt", "w");
    fprintf(f, "%d\n", n);  // Ghi ra so sinh vien
    for (i = 0; i < n; i++)
        fprintf(f, "%-25s%-9d%.2f\n", ht[i], ns[i], dtb[i]); // Ghi ra thong tin tren mot dong
    fclose(f); // Ket thuc viec ghi ra thong tin sinh vien bang cach dong file
    return 0;
}
```

Màn hình:

```
Nhap vao so sv: 3
Nhap thong tin cua 3 sv:
Ho va ten:  Cong Phuong Cao
Nam sinh:   1996
Diem tb:    9.99
Ho va ten:  Nguyen Thi Kieu Trang
Nam sinh:   1997
Diem tb:    9.97
Ho va ten:  La Thi Ngoc Mai
Nam sinh:   1996
Diem tb:    9.98
```

File *data_output.txt*:

```
3
Cong Phuong Cao           1996     9.99
Nguyen Thi Kieu Trang     1997     9.97
La Thi Ngoc Mai           1996     9.98
```

Lưu ý: Cấu trúc:

- Mảng cho phép ta xác định kiểu dữ liệu của biến để có thể lưu các mục dữ liệu **cùng loại**. Cấu trúc là một kiểu dữ liệu **khác** có sẵn trong C, do người dùng tự định nghĩa, cho phép lưu các mục dữ liệu thuộc các kiểu dữ liệu **khác loại**.
- Ta có thể truyền một cấu trúc dưới dạng đối số hàm theo cách giống như cách ta truyền bất kỳ biến hoặc con trỏ nào khác.
- Ta có thể xác định con trỏ trỏ tới cấu trúc giống như cách ta xác định con trỏ trỏ tới bất kỳ biến nào khác. Để truy cập các thành phần của một cấu trúc bằng cách sử dụng một con trỏ trỏ tới cấu trúc đó, ta phải sử dụng toán tử "`->`", thay vì toán tử "`.`".

Lưu ý: File:

- Cần nhớ thực hiện thao tác đóng file (sử dụng lệnh *fclose*) khi đã hoàn tất quá trình đọc hoặc/và ghi file.
- Khi đọc vào dữ liệu từ file, cần nhớ lưu file dữ liệu đầu vào trước khi chạy chương trình. Ngoài ra, nếu chương trình không chỉ rõ đường dẫn đến file, file cần nằm trong cùng một folder với chương trình chạy.
- Khi ghi ra dữ liệu vào file, nếu trong máy tính chưa tồn tại file đó, chương trình sẽ tự tạo ra một file mới với tên tương ứng để thực hiện việc ghi dữ liệu.
- Đặc tả "`%25[^\n]`" ở lệnh *scanf/fscanf* có nghĩa là đọc vào các ký tự ngoại trừ dấu xuống dòng và đọc vào không quá 25 ký tự.
- Đặc tả "`%-25s`" ở lệnh *printf/fprintf* có nghĩa là in ra tối đa 25 ký tự của một xâu ký tự, và xâu ký tự in ra được dịch về phía bên trái (nếu xâu ít hơn 25 ký tự).

Bài tập

File:

1. Áp dụng ví dụ 3 và ví dụ 4, hãy viết một chương trình đọc vào dữ liệu sinh viên từ file ("data_input.txt"), rồi ghi ra những dữ liệu vừa đọc được vào file ("data_output.txt").

Input	Output
3 Cong Phuong Cao 1996 9.99 Nguyen Thi Kieu Trang 1997 9.97 La Thi Ngoc Mai 1996 9.98	3 Cong Phuong Cao 1996 9.99 Nguyen Thi Kieu Trang 1997 9.97 La Thi Ngoc Mai 1996 9.98

2. Áp dụng code của bài tập 1, hãy viết một chương trình đọc vào dữ liệu sinh viên từ file ("data_input.txt"), rồi ghi ra những dữ liệu vừa đọc được vào những file khác nhau:
   - Dữ liệu những sinh viên có điểm trung bình ≥ 8.5 ghi vào file "data_student_A.txt"
   - Dữ liệu những sinh viên có điểm trung bình ≥ 7.0 và < 8.5 ghi vào file "data_student_B.txt"
   - Dữ liệu những sinh viên có điểm trung bình ≥ 5.5 và < 7.0 ghi vào file "data_student_C.txt"
   - Dữ liệu những sinh viên có điểm trung bình ≥ 4.0 và < 5.5 ghi vào file "data_student_D.txt"
   - Dữ liệu những sinh viên có điểm trung bình < 4.0 ghi vào file "data_student_F.txt"

3. Cho một mảng có n phần tử. Tìm giá trị nhỏ nhất và lớn nhất của mảng đó.
   Input được đọc vào từ một file tên là *data3.txt*.
   Input
   dòng thứ nhất là số nguyên n
   dòng thứ hai là n số nguyên
   Output
   2 số nguyên là giá trị nhỏ nhất và lớn nhất của mảng

Input	Output
7 5 3 6 2 7 1 4	1 7

4. Cho một mảng có n phần tử. Sắp xếp lại các phần tử của mảng đó theo thứ tự tăng dần.
   Input được đọc vào từ một file tên là *data4.txt*.
   Output được ghi ra một file tên là *outp4.txt*.

Input
dòng thứ nhất là số nguyên n
dòng thứ hai là n số nguyên
Output
n số là các phần tử của mảng sau khi đã sắp xếp

Input	Output
7 5 3 6 2 7 1 4	1 2 3 4 5 6 7

5. Cho ma trận a có m hàng và n cột, và ma trận b có p hàng q cột. Xét xem hai ma trận a và b có thực hiện được phép nhân ma trận (đại số tuyến tính) không, nếu có thì thực hiện phép toán và in ra kết quả, nếu không thì in ra dòng chữ "INVALID".
Input được đọc vào từ một file tên là *data5.txt*.
Output được ghi ra một file tên là *outp5.txt*.

Input
dòng đầu là 2 số m và n
m dòng tiếp, mỗi dòng chứa n số là các phần tử ma trận a
dòng tiếp là 2 số p và q
p dòng tiếp, mỗi dòng chứa q số là các phần tử ma trận b
Output
đáp án của bài toán

Input	Output
1 2 2 1 2 3 1 2 3 4 5 6	6 9 12

6. **Lập bảng các số nguyên tố nhỏ hơn 100**

Trước hết ta viết các số tự nhiên từ 2 đến 99, chúng gồm các số nguyên tố và hợp số. Ta sẽ loại đi các hợp số. Ta đã biết các số nguyên tố nhỏ hơn 10 là 2, 3, 5, 7.
Giữ lại số 2, loại các số là bội của 2 mà lớn hơn 2.
Giữ lại số 3, loại các số là bội của 2 mà lớn hơn 3.
Giữ lại số 5, loại các số là bội của 2 mà lớn hơn 5.
Giữ lại số 7, loại các số là bội của 2 mà lớn hơn 7.
Các số còn lại trong bảng không chia hết cho mọi số nguyên tố nhỏ hơn 10. Chúng là các số nguyên tố và được đóng khung trong bảng sau: (xem bảng bên cạnh)
Ta được 25 số nguyên tố nhỏ hơn 100(*) là: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97.
Số nguyên tố nhỏ nhất là số 2, đó là số nguyên tố chẵn duy nhất.

		2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69
70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89
90	91	92	93	94	95	96	97	98	99

Hình 1

(*) Trong cách làm trên, các hợp số được sàng lọc đi, các số nguyên tố được giữ lại. Nhà toán học cổ Hi Lạp Ơ-ra-tô-xten (276 - 194 trước Công nguyên) đã viết các số trên giấy cỏ sậy căng trên một cái khung rồi dùi thủng các hợp số. Bảng các số nguyên tố còn lại giống như một cái sàng và được gọi là sàng Ơ-ra-tô-xten.

Hãy viết một chương trình tìm các số nguyên tố nhỏ hơn 100 rồi in ra thành bảng tương tự như *Hình 1* để tạo thành sàng Eratosthenes (*Eratosthenes's sieve*).
Đối với các số là hợp số, in ra các dấu gạch dưới ('_') ứng với số chữ số của số đó.
Kết quả được ghi ra một file tên là *outp6.txt*.
☞ Mở rộng bài toán: In ra sàng Eratosthenes cho các số nguyên tố nhỏ hơn 400, 625, 900.

Output

```
__ __  2  3 __  5 __  7 __ __
__ 11 __ 13 __ __ __ 17 __ 19
__ __ __ 23 __ __ __ __ __ 29
__ 31 __ __ __ __ __ 37 __ __
__ 41 __ 43 __ __ __ 47 __ __
__ __ __ 53 __ __ __ __ __ 59
__ 61 __ __ __ __ __ 67 __ __
__ 71 __ 73 __ __ __ __ __ 79
__ __ __ 83 __ __ __ __ __ 89
__ __ __ __ __ __ __ 97 __ __
```

Cấu trúc:

7. Cấu trúc: *sinhvien*

   (a) Áp dụng ví dụ 2, ví dụ 3 và ví dụ 4, hãy viết một chương trình sử dụng cấu trúc *sinhvien* đọc vào dữ liệu sinh viên từ file ("data_input.txt"), rồi ghi ra những dữ liệu vừa đọc được vào file ("data_output.txt").

Input	**Output**
3 Cong Phuong Cao 1996 9.99 Nguyen Thi Kieu Trang 1997 9.97 La Thi Ngoc Mai 1996 9.98	3 Cong Phuong Cao 1996 9.99 Nguyen Thi Kieu Trang 1997 9.97 La Thi Ngoc Mai 1996 9.98

   (b) Tiếp tục câu trước, bổ sung thêm thêm các trường thông tin sau vào cấu trúc *sinhvien*: *điểm chuyên cần*, *điểm giữa kỳ*, và *điểm cuối kỳ*. Từ đó, hãy tính lại *điểm trung bình* dựa trên công thức:

$$\textit{điểm trung bình} = 20\%\textit{điểm chuyên cần} + 20\%\textit{điểm giữa kỳ} + 60\%\textit{điểm cuối kỳ}$$

   Sử dụng file để thực hiện các thao tác đọc vào và in ra!

   Input
   dòng 1 là số sinh viên n
   n dòng tiếp, mỗi dòng chứa họ và tên sinh viên (25 ký tự đầu dòng), năm sinh, điểm chuyên cần, điểm giữa kỳ, và điểm cuối kỳ
   Output
   dòng 1 là số sinh viên n
   n dòng tiếp, mỗi dòng chứa họ và tên sinh viên (25 ký tự đầu dòng), năm sinh, và điểm trung bình

Input	**Output**
5 Cong Phuong Cao 1996 9 9 7 Nguyen Thi Kieu Trang 1997 9 9 9 La Thi Ngoc Mai 1996 8 7 10 Nguyen Bach Duong 1996 6 8 10 Dinh Dam Khanh 1996 7 6 8	5 Cong Phuong Cao 1996 7.80 Nguyen Thi Kieu Trang 1997 9.90 La Thi Ngoc Mai 1996 9.00 Nguyen Bach Duong 1996 8.80 Dinh Dam Khanh 1996 7.40

   (c) Tiếp tục câu trước, sắp xếp các *sinhvien* theo chiều giảm dần của điểm trung bình, rồi in ra kết quả. Sử dụng file để thực hiện các thao tác đọc vào và in ra!

   Input
   dòng 1 là số sinh viên n
   n dòng tiếp, mỗi dòng chứa họ và tên sinh viên (25 ký tự đầu dòng), năm sinh, và điểm trung bình
   Output
   dòng 1 là số sinh viên n
   n dòng tiếp, mỗi dòng chứa họ và tên sinh viên (25 ký tự đầu dòng), năm sinh, và điểm trung bình

Input	**Output**
5 Cong Phuong Cao 1996 7.80 Nguyen Thi Kieu Trang 1997 9.90 La Thi Ngoc Mai 1996 9.00 Nguyen Bach Duong 1996 8.80 Dinh Dam Khanh 1996 7.40	5 Nguyen Thi Kieu Trang 1997 9.90 La Thi Ngoc Mai 1996 9.00 Nguyen Bach Duong 1996 8.80 Cong Phuong Cao 1996 7.80 Dinh Dam Khanh 1996 7.40

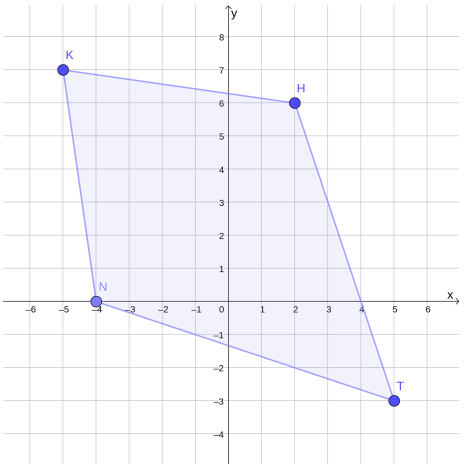
8. Cấu trúc: *point*

   (a) Tạo một cấu trúc *point* chứa các (trường) thông tin của một điểm trong không gian 2D bao gồm hoành độ và tung độ. Cho 2 điểm $A(x_1, y_1)$, $B(x_2, y_2)$ (hai biến A và B có kiểu dữ liệu *point*). Tính độ dài đoạn thẳng AB. Kết quả ở màn hình có 4 chữ số thập phân.

Input
dòng thứ nhất gồm 2 số x_1 và y_1 là tọa độ điểm A
dòng thứ hai gồm 2 số x_2 và y_2 là tọa độ điểm B
Output
1 số duy nhất là độ dài đoạn thẳng AB

Input	Output
1 1 2 3	2.2361

(b) Tiếp tục câu trước, cho một đa giác lồi có n đỉnh (n điểm có kiểu dữ liệu *point*) có tọa độ (x_i, y_i) với $i = 1..n$. Tính chu vi đa giác.



Input
dòng 1 là 1 số n
n dòng tiếp mỗi dòng có 2 số x_i, y_i là tọa đỉnh i của đa giác
Output
1 số duy nhất là chu vi đa giác

Input	Output
4 -5 7 2 6 5 -3 -4 0	33.115802

☞ Mở rộng bài toán:
+ Tạo cấu trúc *polygon*, mô tả một đa giác lồi (xác định bởi tập hợp của các *point*), chứa các thông tin của một đa giác trong không gian 2D.
+ Tạo cấu trúc *circle*, mô tả một đường tròn (xác định bởi một point và bán kính), chứa các thông tin của một đường tròn trong không gian 2D.
+ Tạo cấu trúc *line*, mô tả một đường thẳng, chứa các thông tin của một đường thẳng trong không gian 2D. Thông thường, *line* có thể được xác định bởi ba số a, b, và c của phương trình đường thẳng: $ax + by + c = 0$.
+ Lập các hàm: tính góc của được tạo nên bởi ba điểm; tính hướng rẽ sang phải hoặc trái (ccw - *counterclockwise turns*) của 3 điểm $A \to B \to C$; tính điểm giao cắt giữa *line* với *line*, giữa *line* với *circle*, giữa *circle* với *circle*, giữa *line* với *polygon*, ...

9. Cấu trúc: *matrix*

(a) Tạo một cấu trúc *matrix* chứa các (trường) thông tin của một ma trận, bao gồm giá trị các phần tử, kích thước hàng và kích thước cột. Cho một ma trận có kích thước m hàng và n cột. Lập các hàm thực hiện thao tác đọc vào và ghi ra *matrix* khi biết kích thước m và n. Sử dụng file để thực hiện các thao tác đọc vào và in ra!
Input
dòng đầu là 2 số m và n
m dòng tiếp, mỗi dòng chứa n số là các phần tử ma trận
Output
m dòng, mỗi dòng chứa n số là các phần tử ma trận

Input	Output
2 3 1 2 3 4 5 6	1 2 3 4 5 6

(b) Tiếp tục câu trước, cho *matrix* A có m hàng và n cột, và *matrix* B có p hàng q cột. Lập hàm xét xem hai *matrix* A và B có thực hiện được phép nhân ma trận không. Hàm này trả về 2 đối số đầu ra: *flag* và C, trong đó *flag* có giá trị 0 hoặc 1 tương ứng với các trường hợp không thực hiện được hoặc có thực hiện được phép nhân ma trận; C là kết quả của phép nhân ma trận, trong trường hợp không thực hiện được phép nhân, $C = NULL$. Sử dụng file để thực hiện các thao tác đọc vào và in ra!
Input
dòng đầu là 2 số m và n
m dòng tiếp, mỗi dòng chứa n số là các phần tử *matrix* A
dòng tiếp là 2 số p và q
p dòng tiếp, mỗi dòng chứa q số là các phần tử *matrix* B
Output
dòng 1 là giá trị của *flag*
nếu *flag* $\neq 0$, các dòng tiếp theo là các phần tử *matrix* C

Input	Output
1 2 2 1 2 3 1 2 3 4 5 6	1 6 9 12

(c) Tiếp tục câu trước, cho *matrix* A có kích thước m hàng và n cột. Lập hàm thực hiện thuật toán khử Gauss trên *matrix* A rồi in ra kết quả. Sử dụng file để thực hiện các thao tác đọc vào và in ra!

Input
dòng đầu là 2 số m và n
m dòng tiếp, mỗi dòng chứa n số là các phần tử *matrix* A
Output
ma trận A sau khi khử Gauss

Input	Output
3 4 2 1 -1 8 -3 -1 2 -11 -2 1 2 -3	2.0 1.0 -1 8 0 0.5 0.5 1 0 0 -1 1

(d) Tiếp tục câu trước, cho *matrix* vuông A có kích thước n. Lập hàm tính ma trận nghịch đảo của A. Hàm này trả về 2 đối số đầu ra: *flag* và A^{-1}, trong đó *flag* có giá trị 0 hoặc 1 tương ứng với các trường hợp ma trận A không khả đảo hoặc khả đảo; *matrix* A^{-1} là ma trận nghịch đảo của A, trong trường hợp A không khả đảo, $A^{-1} = NULL$. Sử dụng file để thực hiện các thao tác đọc vào và in ra!
Input
dòng đầu là 1 số n
n dòng tiếp, mỗi dòng chứa n số là các phần tử *matrix* A
Output
dòng 1 là giá trị của *flag*
nếu *flag* $\neq 0$, các dòng tiếp theo là các phần tử *matrix* A^{-1}
(các giá trị của A^{-1} làm tròn đến chữ số thứ 2 sau dấu phẩy)

Input	Output
3 1 0 4 1 1 6 -3 0 -10	-5.00 0.00 -2.00 -4.00 1.00 -1.00 1.50 0.00 0.50

☞ Mở rộng bài toán: Lập các hàm khác xử lý và tính toán trên ma trận sử dụng cấu trúc *matrix*. Tham khảo tờ thực hành về nội dung **Hàm**.

10. Cấu trúc: *bignum*

(a) Tạo một cấu trúc *bignum* chứa các (trường) thông tin của một số lớn (số có nhiều chữ số), bao gồm giá trị các chữ số, dấu, và số chữ số. Cho một số lớn a. Lập hàm tính số chữ số của một số lớn cho trước mà sử dụng cấu trúc *bignum*. Sử dụng file để thực hiện các thao tác đọc vào và in ra!
Input
1 là xâu ký tự mô tả số lớn a
Output
số chữ số của số lớn a

Input	Output
123456789101112131415 16	23

(b) Tiếp tục câu trước, cho một số lớn a. Lập hàm thực hiện thao tác đọc vào một số lớn và ghi ra từng chữ số của số lớn đó mà sử dụng cấu trúc *bignum*.
Input
1 là xâu ký tự mô tả số lớn a
Output
từng chữ số của số lớn a (kiểu *int*, cách nhau bởi dấu cách)

Input	Output
123456789101112131415 16	1 2 3 4 5 6 7 8 9 1 0 1 1 1 2 1 3 1 4 1 5 1 6

☞ Gợi ý: :
+ `int d = '7' - '0'; // d = 7`
+ `char c = 8 + '0'; // c = '8'`

(c) Tiếp tục câu trước, cho hai số lớn a và b. Lập hàm thực hiện thao tác so sánh hai số a và b. Hàm trả về các giá trị 1, 0, hoặc -1 tương ứng với các trường hợp $a > b$, $a = b$, hoặc $a < b$.
Input
dòng 1 là xâu ký tự mô tả số lớn a
dòng 2 là xâu ký tự mô tả số lớn b
Output
1 số là kết quả của phép so sánh

Input	Output
987654321 876543210	1

Input	Output
99999888887777766666555554444 4 99989796959493929190898886858483	-1

(d) Tiếp tục câu trước, cho hai số lớn a và b. Lập hàm thực hiện thao tác tính tổng $a + b$.
Input
dòng 1 là xâu ký tự mô tả số lớn a
dòng 2 là xâu ký tự mô tả số lớn b
Output
xâu ký tự mô tả tổng $a + b$

Input	Output
987654321 876543210	1864197531

Input	Output
99999888887777766666555554444499998979695949392919089888786858483	99999796948382706967565543424029 27

☞ Mở rộng bài toán: Áp dụng phép toán cộng *bignum* để tính số Fibonacci thứ 200, F_{200}.

(e) Tiếp tục câu trước, cho hai số lớn a và b. Lập hàm thực hiện thao tác tính tích $a \times b$.

Input
dòng 1 là xâu ký tự mô tả số lớn a
dòng 2 là xâu ký tự mô tả số lớn b

Output
xâu ký tự mô tả tổng $a \times b$ (1 dòng)

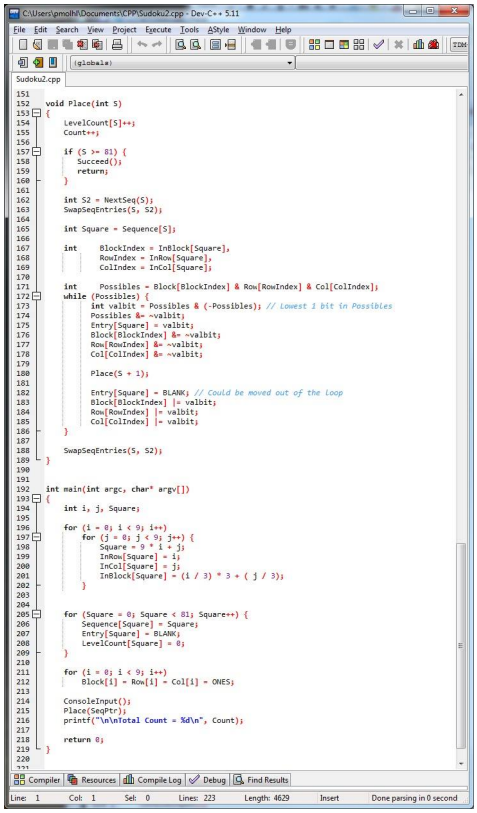
Input	Output
987654321 876543210	865721688899710410

Input	Output
99999888887777766666555554444499998979695949392919089888786858483	99989685858608520899116869049599012226312640525536863254449184 52

☞ Mở rộng bài toán:
+ Ngoài phép *cộng* *bignum* và *nhân* *bignum*, tìm cách thực hiện các phép toán tương tự: *trừ* *bignum* và *chia* *bignum*.
+ Để tăng tốc độ thực hiện cũng như tiết kiệm bộ nhớ, sử dụng cơ số 1,000,000,000 thay vì cơ số 10.

11. **Khả năng lập trình của Thủ tướng Singapore Lý Hiển Long** (trích *VnExpress*)

Thủ tướng Singapore Lý Hiển Long từng đăng mã nguồn một chương trình do ông tự viết và được giới trong ngành đánh giá tốt (Hình 2). Thủ tướng Singapore tháng 5/2015 chia sẻ trên facebook mã nguồn chương trình giải Sudoku mà ông viết. Ông nói rằng chương trình khá đơn giản và ông viết nó bằng ngôn ngữ C++. Bài đăng của ông thu hút 52.000 lượt thích và hơn 2.000 bình luận.
Sudoku là trò chơi điền số từ 1 đến 9 vào những ô trống sao cho mỗi cột dọc, mỗi hàng ngang, mỗi phân vùng nhỏ có đủ các số từ 1 đến 9 mà không được lặp lại.
Hai giảng viên đại học đã khen ngợi khả năng viết code (dãy các câu lệnh được viết bằng ngôn ngữ lập trình) của Thủ tướng Lý Hiển Long. Giảng viên Ng Wee Keong, phó giám đốc nghiên cứu của trường công nghệ máy tính thuộc Đại học Công nghệ Nanyang, nói rằng code này "có cấu trúc tốt" trong khi đó, giảng viên Aaron Tan của trường máy tính thuộc Đại học Quốc gia Singapore nói rằng nó "được viết tốt".
"Nó cho thấy phương pháp thiết kế và sự tinh thông logic của người lập trình", ông Ng nói, nhấn mạnh đoạn code này rành mạch và dễ hiểu.
Tuy nhiên, cả hai chuyên gia đều nói rằng mã nguồn này được viết bằng ngôn ngữ lập trình C chứ không phải C++ như ông Lý nói. Giải thích lý do ông Lý có thể nói nó được viết bằng C++, ông Tan cho biết: "Không sai khi nói rằng đó là một chương trình C++ (vì nó hoạt động tốt với trình biên dịch C++), nhưng nó được viết theo phong cách C".



Hình 2: Mã nguồn Thủ tướng Singapore Lý Hiển Long chia sẻ. Ảnh: Facebook

Ông Ng nói rằng tất cả sinh viên khoa học máy tính và kỹ thuật tại trường ông phải học ngôn ngữ lập trình cơ bản này trong năm học đầu tiên. Ông nói thêm rằng sẽ mất một hoặc hai khóa học về lập trình cấu trúc để viết một chương trình giải Sudoku tương tự.
Một số người bình luận trong bài viết của Thủ tướng Lý cho biết họ rất ngạc nhiên khi ông viết code, nhưng những người khác thì cho rằng điều này không bất ngờ vì ông Lý là một sinh viên toán xuất sắc tại Đại học Cambridge và còn có văn bằng khoa học máy tính tương đương thạc sĩ.
Nhiều người đề nghị ông tham dự nhiều sự kiện công nghệ, số khác thì cho rằng kỹ năng viết code cho thấy ông Lý có khả năng giải quyết vấn đề tốt trong tình huống hỗn loạn.

"Tôi thấy một nhà lãnh đạo với khả năng suy nghĩ có tính hệ thống, tổ chức tốt, đưa mọi thứ vào trật tự. Thật ấn tượng!", người tên Chou Chung bình luận.

Hình 3: Thủ tướng Singapore Lý Hiển Long. Ảnh: Time

Bài tập lập trình cuối cùng của môn học rất đơn giản: Hãy viết một chương trình giải Sudoku dựa trên tất cả các kiến thức đã học. Sử dụng file để thực hiện các thao tác đọc vào và in ra!
Lưu ý: Trong trường hợp có nhiều đáp án đúng, chỉ cần chỉ ra duy nhất 1 đáp án.

Input
9 dòng, mỗi dòng có 9 số từ 0 đến 9 là các giá trị các ô trên bảng Sudoku, số 0 là ô giá trị chưa xác định

Output
9 dòng, mỗi dòng có 9 số là các giá trị các ô trên bảng Sudoku sau khi đã giải xong

Input	**Output**
0 7 0 0 2 0 1 0 0 0 0 5 3 0 0 7 8 0 1 3 0 0 0 0 0 0 0 0 4 0 2 9 0 0 0 0 0 0 6 0 0 0 9 0 0 9 0 0 0 6 8 0 7 0 0 0 0 0 0 0 0 1 8 0 8 1 0 0 6 3 0 0 0 0 3 0 4 0 0 5 0	6 7 8 5 2 9 1 3 4 2 9 5 3 1 4 7 8 6 1 3 4 6 8 7 2 9 5 3 4 7 2 9 5 8 6 1 8 5 6 1 7 3 9 4 2 9 1 2 4 6 8 5 7 3 5 6 9 7 3 2 4 1 8 4 8 1 9 5 6 3 2 7 7 2 3 8 4 1 6 5 9

Input	**Output**
0 0 0 0 0 0 0 0 0 0 0 4 9 0 7 1 0 0 0 7 8 5 0 1 6 4 0 9 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 5 0 0 0 0 0 0 0 7 0 3 9 1 0 4 5 2 0 0 0 1 7 0 5 3 0 0 0 0 0 0 0 0 0 0 0	1 9 2 3 4 6 7 8 5 6 5 4 9 8 7 1 3 2 3 7 8 5 2 1 6 4 9 9 2 7 6 5 8 4 1 3 8 1 3 4 7 9 2 5 6 5 4 6 2 1 3 8 9 7 7 3 9 1 6 4 5 2 8 2 8 1 7 9 5 3 6 4 4 6 5 8 3 2 9 7 1

Giải thích: Test ví dụ 1 có duy nhất 1 đáp án đúng. Test ví dụ 2 có 501 đáp án đúng và Output mẫu là 1 đáp án đúng trong số đó.