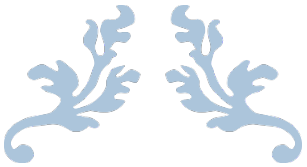
ĐẠI HỌC NGOẠI NGỮ - TIN HỌC TP.HCM
KHOA CÔNG NGHỆ THÔNG TIN



KỸ THUẬT LẬP TRÌNH

BÀI TẬP THỰC HÀNH

THÁNG 3, 2019

NỘI DUNG

C# CODING STANDARDS

Coding standards có mục đích: tạo định dạng nhất quán trong quá trình viết code, giúp người đọc chương trình tập trung vào nội dung chứ không mất thời gian dò tìm tên biến, tên hàm, các kiểu dữ liệu, … từ đó đọc hiểu code nhanh hơn cũng như tái sử dụng, nâng cấp, bảo trì code nhanh chống. Trong phần này chúng ta điểm qua:

- Coding style
- Naming conventions

Layout Conventions (Bố cục)

- Viết chỉ một lệnh trên một dòng
- Viết chỉ một khai báo trên một dòng
- Nếu các dòng tiếp tục không được thụt lề tự động, hãy thụt lề một tab stop (bốn dấu cách)
- Dùng các dấu ngoặc để tạo các mệnh đề trong biểu thức rõ ràng hơn

```
if ((val1 > val2) && (val1 > val3))
{
}
```

Comment

- Đặt comment trên một dòng riêng (không phải cuối dòng code)
- Bắt đầu comment text bằng ký tự viết hoa
- Kết thúc comment là dấu chấm
- Insert một khoảng trắng giữa // với text

```
// The following declaration creates a query. It does not run
// the query.
```

Thuật ngữ

- Camel Case (**camelCase**): Ký tự đầu tiên của tên viết thường và mỗi từ sau đó viết hoa ký tự đầu tiên. Ví dụ: **s**tudent**N**ame
- Pascal Case (**PascalCase**): Ký tự đầu tiên của mỗi từ được viết hoa. Ví dụ: **G**et**P**ost

Class

- Dùng **PascalCase** cho tên lớp
- Dùng **danh từ hay cụm danh từ** cho tên lớp

```
public class Customer
{
}

public class Image
{
}
```

Methods

- Dùng **PascalCase** cho tên phương thức
- Dùng **động từ** cho tên phương thức

```
public int TransferAccount(string accountNumber)
{
}
```

Tham số của hàm và biến cục bộ

- Dùng **camelCase** cho tên tham số của hàm và biến cục bộ

```
public int TransferAccount(string accountNumber)
{
}
```

Property

- Dùng **PascalCase** cho tên property
- Không dùng Get/Set làm prefix của tên property

```
private int salary = 1000;
public string Salary
{
  get
  {
    return salary;
  }

  set
  {
    salary = value;
  }
}
```

Interface

- Dùng ký tự "**I**" làm prefix của tên interface
- Sau "**I**" đặt tên theo định dạng **PascalCase**

```
public interface IUser
{
  bool ValidateUser();
}
```

Namespace

- Dùng PascalCase cho tên namespace

Các kiểu dữ liệu cơ bản

Các kiểu dữ liệu cơ bản thường hay được sử dụng trong lập trình cơ bản là

- Scalar
- Danh sách (mảng một chiều)
- Bảng (matrix hay mảng hai chiều)

Scalar

Dữ liệu	Miền giá trị
int	-2 tỷ đến 2 tỷ
double	$-1.79 \times 10^{-308} \rightarrow 1.79 \times 10^{308}$
char	Ký tự ($0 \rightarrow 65535$)
string	Chuỗi

```
double score;
int num;
string name;
```

Danh sách

Dãy n phần tử $a = (a_0, a_1, \dots, a_{n-1})$ chứa các

- Số nguyên, số thực: dãy số
- Chuỗi (string): danh sách họ tên sinh viên, danh sách sản phẩm, …

```
double[] scores;
int num;

num = …

scores = new int[num];
```

```
string[] names;
int num;

num = …

names = new string[num];
```

Bảng

- Dữ liệu dạng bảng (hay còn gọi là ma trận) chứa một bảng các số nguyên, số thực, …
- Ký hiệu: $a[m\ x\ n]$

```
int[,] a;
int m, n;

m = …
n = …

a = new int[m, n];
…
```

Các câu lệnh cơ bản

Các lệnh cơ bản của ngôn ngữ lập trình C#

- Nhập xuất dữ liệu, Xuất số lẻ
- if
- for
- while
- foreach
- do…while

Nhập/Xuất dữ liệu

- Nhập mỗi dòng một giá trị

```
int inputInt = Convert.ToInt32(Console.ReadLine());
double inputDouble = Convert.ToInt32(Console.ReadLine());
```

- Nhập mỗi dòng nhiều giá trị

```
string[] tokens = Console.ReadLine().Split();

int a = Convert.ToInt32(tokens[0]);
int b = Convert.ToInt32(tokens[1]);
```

- Xuất dữ liệu

```
Console.Write(inputInt);
Console.Write(inputInt + “ ” + inputDouble);
Console.Write("Num {0} {1}", inputInt, inputDouble);
Console.Write($"Text {inputInt} {inputDouble}");
Console.WriteLine();
```

- Xuất số lẻ

```
Console.Write("Text {0:0.00}", num2);
```

Câu lệnh rẽ nhánh if

```
if (điều kiện) {
}
```

```
if (điều kiện) {
}
else {
}
```

- Phép toán logic
  - So sánh: >, <, >=, <=, ==, !=
  - Kết hợp: &&, ||, !

Câu lệnh rẽ nhánh switch

```
switch (điều kiện)
{
  case value1:
    …
    break;

  default:
}
```

Câu lệnh lặp

```
for (i=0; i<n; i++) {

}
```

```
while (điều kiện) {

}
```

```
foreach (var item in collection) {

}
```

```
do {

} while (điều kiện)
```

Methods

```
public static DataType MethodName(parameters)
{
}
```

Lớp

```
class ClassName
{
}
```

Kỹ thuật debug

Trong phần này chúng ta sẽ điểm qua các kỹ thuật debug cơ bản nhất được sử dụng trong quá trình viết chương trình, cụ thể:

- Quy trình debug cơ bản
- Một số debugger tool trong Visual studio

Kiểm thử (testing) chương trình với một các dữ liệu input được thiết kế tốt giúp cho lập trình viên tự tin rằng chương trình mình viết ra là đúng đắn. Trong quá trình kiểm thử chương trình, lập trình viên quan sát mối quan hệ input – ouput. Nếu chương trình sinh ra output như mong đợi thì chương trình được kiểm thử thành công, ngược lại, chương trình được cho là có lỗi (error, bug, defect). Trong tình huống này, kiểm thử là cách giúp cho chúng ta phát hiện lỗi trong chương trình, nhưng không cho chúng ta biết nguyên nhân gây lỗi và cách sửa code để khắc phục lỗi như thế nào. Để tìm nguyên nhân gây lỗi và cách khắc phục lỗi, lập trình viên phải qua giai đoạn debug.

Debug là quá trình tìm và giải quyết các lỗi trong chương trình. Tiến trình debug thường trải qua các giai đoạn sau:

1. Tái sinh lỗi, xác định input nào gây ra lỗi
2. Dùng debugger tool để
    a. Kiểm tra trạng thái của chương trình (giá trị các biến, stack track)
    b. Truy ra nguồn gốc vấn đề
3. Trong một số trường hợp đơn giản, chúng ta có thể tracing bằng cách in các giá trị của các biến tại một số nơi trong code để phát hiện vấn đề

Các debugger tool trong Visual Studio

Bước 1. Tạo breakpoint (điểm dừng)

- Xác định dòng chương trình muốn bắt đầu debug từ đó
- Tạo breakpoint tại điểm đã xác định: ***F9***

Bước 2. Chạy debug đến điểm breakpoint

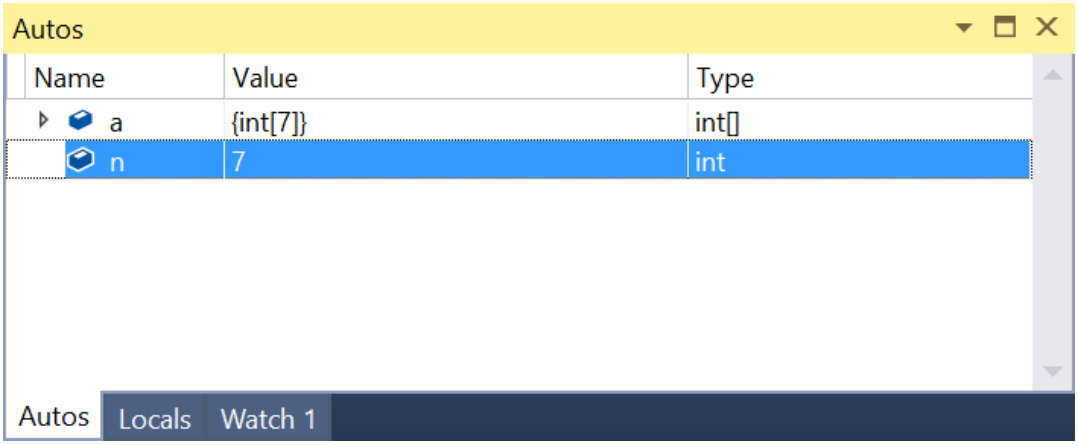
- Nhấn ***F5***

Bước 3. Chạy đến lệnh kế tiếp

- Step Into: ***F11 (vào bên trong lời gọi hàm)***
- Step Over: ***F10 (bỏ qua lời gọi hàm)***
- Step Out: ***Shift + F11***

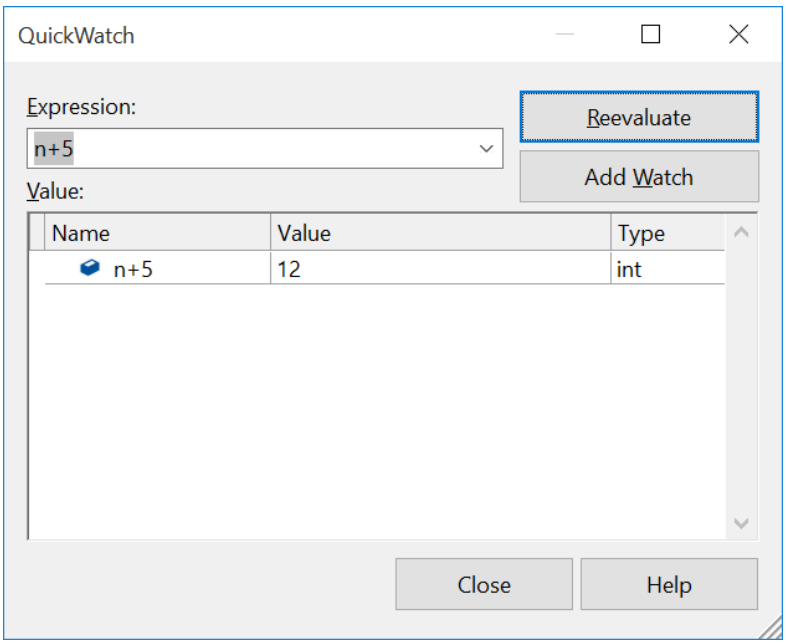
Bước 4. Quan sát các giá trị của các biến

- Cửa sổ Autos: Cho chúng ta thấy các biến, các giá trị hiện tại của và kiểu của biến



Cửa sổ Autos

- QuickWatch: ***Shift + F9***
  - Dùng để tính toán giá trị của biểu thức trong quá trình debug



Cửa sổ QuickWatch

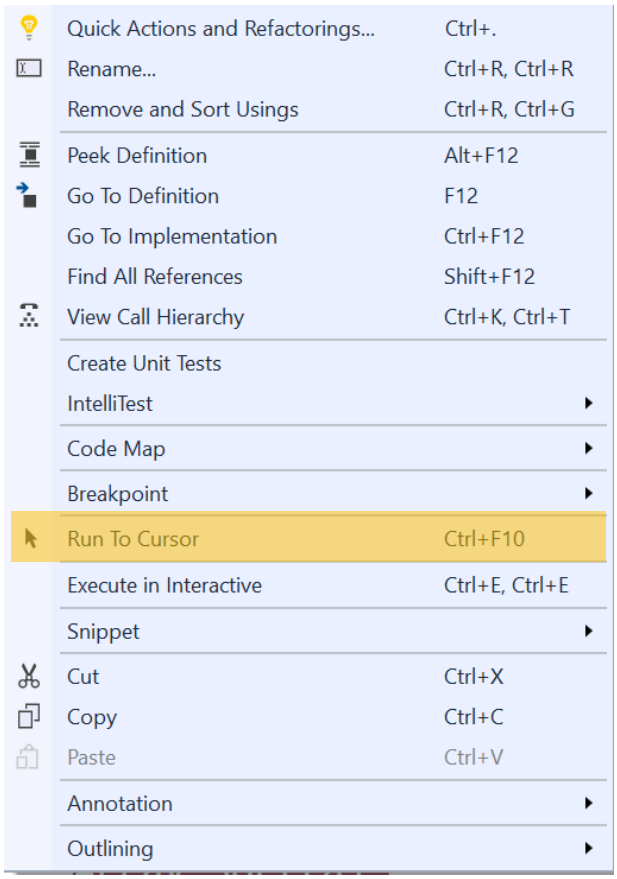
Bước 5. Tắt chế độ debug

- Nhấn: ***Shift + F5***

Chức năng khác

Run to Cursor:

- Click phải tại nơi muốn chạy đến → Run To Cursor



Chức năng Run to Cursor

Restart debug

- Nhấn nút trong Debug Toolbar hay
- Ctrl + Shift +F5

Buổi 1. Dữ liệu dạng bảng

Sau khi hoàn thành bài thực hành này sinh viên có thể:

- Khai báo, tạo, nhập và xuất dữ liệu dạng bảng
- Sử dụng bảng để lưu trữ và tính toán

Bài 1. In bảng số

Cho bảng số nguyên $a[m \times n], (1 \leq m, n \leq 100)$. Hãy viết chương trình in ra các dòng và in ra các cột như sau:

In ra các dòng

$\boldsymbol{Row} < space > \boldsymbol{i}: < space > \boldsymbol{num1} < space > \boldsymbol{num2}$ ...

In ra các cột

$\boldsymbol{Col} < space > \boldsymbol{i}: < space > \boldsymbol{num1} < space > \boldsymbol{num2}$ ...

Ví dụ: Giả sử mảng $a[2x3]$ có các giá trị sau

$$a = \begin{pmatrix} 4 & 5 & 6 \\ 12 & 13 & 14 \end{pmatrix}$$

In mảng theo từng hàng

```
Row 0: 4 5 6
Row 1: 12 13 14
```

In mảng theo từng cột

```
Col 0: 4 12
Col 1: 5 13
Col 2: 6 14
```

Input

- Dòng thứ nhất chứa hai số nguyên: m, n
- m dòng tiếp theo, mỗi dòng chưa n số nguyên của bảng a

Output

- Dòng i trong m dòng đầu tiên, mỗi dòng chứa các số của dòng i trong bảng số
  - $Row\ i: num1\ num2\ ...$
- Dòng thứ $m + 1$ chứa dòng trống
- Dòng j trong n dòng tiếp theo, mỗi dòng chứa các số của cột j trong bảng số
  - $Col\ j: num1\ num2\ ...$

Ví dụ

Input	Output
2 3 4 5 6 12 13 14	Row 0: 4 5 6 Row 1: 12 13 14 Col 0: 4 12 Col 1: 5 13 Col 2: 6 14

Bai 2. Nhập xuất bảng số

Cho bảng số nguyên $a[m \times n]$ $(1 \leq m, n \leq 1000)$. Hãy viết chương trình xuất bảng số đã nhập lên màn hình gồm m dòng và n cột.

Input

- Dòng đầu tiên chứa số nguyên m, n là số dòng và số cột của bảng
- m dòng tiếp theo, mỗi dòng chứa n số nguyên

Output

- m dòng, mỗi dòng có n số nguyên

Ví dụ:

Input	Output
3 4 2 3 4 8 8 7 4 4 2 2 4 4	2 3 4 8 8 7 4 4 2 2 4 4

Bài 3. Doanh thu từng ngày

Một cửa hàng bán 3 loại trái cây

- Táo: giá $3
- Cherry: giá $4
- Lê: giá $2

Và số lượng các loại trái cây đã được bán trong 4 ngày như sau

	Thứ 2	Thứ 3	Thứ 4	Thứ 5
Táo	12	15	11	10
Cherry	32	30	40	23
Lê	13	12	20	15

Doanh thu của ngày thứ hai được tính như sau:

$$Số\ tiền\ bán\ loại\ táo\ +\ số\ tiền\ bán\ loại\ cherry\ +\ số\ tiền\ bán\ loại\ Lê$$

$$\$3 \times 12 + \$4 \times 32 + \$2 \times 13 = 190$$

Chúng ta có thể diễn đạt điều này bằng tích chấm (**dot product**) như sau

$$(\$3, \$4, \$2) \cdot (12, 32, 13) = \$3 \times 12 + \$4 \times 32 + \$2 \times 13 = \$190$$

Chúng ta có thể mở rộng kết quả sang cả ma trận

$$(\$3, \$4, \$2) \times \begin{pmatrix} 12 & 15 & 11 & 10 \\ 32 & 30 & 40 & 23 \\ 13 & 12 & 20 & 15 \end{pmatrix} = (\$190, \$189, \$233, \$152)$$

Tổng quát bài toán: một cửa hàng bán m sản phẩm, giá của m sản phẩm được cho trong mảng $a = (a_0, a_1, \dots, a_{m-1})$ với $(1 \le m \le 100)$. Số lượng bán của các sản phẩm trong ngày được cho trong bảng số nguyên $b[m \times n]$ với $(1 \le n \le 1000)$. Trong đó $b_{i,j}$ cho biết số lượng loại sản phẩm i bán ra trong ngày j. Viết chương trình tính doanh thu từng ngày của cửa hàng

Input

- Dòng số đầu tiên chứa hai số nguyên: m, n
- Dòng thứ hai chứa m số nguyên $a_0, a_1, \dots, a_{m-1}$
- m dòng tiếp theo, mỗi dòng chứa n số nguyên của bảng b

Output

- Dòng duy nhất chứa n số, số thứ i là doanh thu của ngày i

Ví dụ

Input	Output
3 4 3 4 2 12 15 11 10 32 30 40 23 13 12 20 15	190 189 233 152

Bài 4. Cộng ma trận

Cho hai bảng số nguyên $a[m \times n], b[m \times n]$ với $(1 \leq m, n, p \leq 100)$. Phép cộng bảng a với bảng b được bảng $c[m \times n]$ có các phần tử $c(i,j)$ được định nghĩa như sau

$$c(i,j) = a(i,j) + b(i,j)$$

Ví dụ

$$a = \begin{pmatrix} 2 & 3 & 4 \\ 5 & 6 & 7 \end{pmatrix}$$

$$b = \begin{pmatrix} 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Mảng kết quả phép cộng

$$c = \begin{pmatrix} 6 & 8 & 10 \\ 12 & 14 & 16 \end{pmatrix}$$

Input

- Dòng số đầu tiên chưa hai số nguyên: m, n
- m dòng tiếp theo, mỗi dòng chứa n số nguyên của bảng a
- m dòng tiếp theo, mỗi dòng chứa n số nguyên là bảng b

Output

- Gồm m dòng, mỗi dòng chứa n số nguyên là bảng tổng của hai bảng trên

Ví dụ

Input	Output
2 3 2 3 4 5 6 7 4 5 6 7 8 9	6 8 10 12 14 16

Bài 5. Nhân ma trận

Cho hai ma trận số nguyên $a[m \times n]$ và $b[n \times p]$ với $(1 \leq m, n, p \leq 100)$. Phép nhân ma trận a với ma trận b được ma trận $c[m \times p]$ có các phần tử $c(i,j)$ được định nghĩa là tích chấm (dot product) của dòng i trong ma trận a với cột j trong ma trận b

$$c(i,j) = \sum_{k=0}^{n-1} a_{i,k} \times b_{k,j}$$

Ví dụ

Ma trận thứ nhất $firstMatrix$

$$firstMatrix = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

Ma trận thứ hai $secondMatrix$

$$secondMatrix = \begin{pmatrix} 5 & 6 & 7 \\ 8 & 9 & 10 \end{pmatrix}$$

Mảng kết quả phép nhân $productMatrix$

$$productMatrix = \begin{pmatrix} 21 & 24 & 27 \\ 47 & 54 & 61 \end{pmatrix}$$

$$productMatrix(0,0) = 1 \times 5 + 2 \times 8 = 21$$
$$productMatrix(0,1) = 1 \times 6 + 2 \times 9 = 24$$
$$productMatrix(0,2) = 1 \times 7 + 2 \times 10 = 27$$

$$productMatrix(1,0) = 3 \times 5 + 4 \times 8 = 47$$
$$productMatrix(1,1) = 3 \times 6 + 4 \times 9 = 54$$
$$productMatrix(1,2) = 3 \times 7 + 4 \times 10 = 61$$

Input

- Dòng số đầu tiên chưa ba số nguyên: m, n, p
- m dòng tiếp theo, mỗi dòng chứa n số nguyên của ma trận a
- n dòng tiếp theo, mỗi dòng chứa p số nguyên của ma trận b

Output

- Gồm m dòng, mỗi dòng chứa p số nguyên là ma trận tích của hai bảng trên

Ví dụ

Input	Output
2 2 1 2 3 4 5 6 7 8 9 10	21 24 27 47 54 61

Bài 6. Chuyển vị ma trận

Cho ma trận số nguyên $a[m \times n]$ với $(1 \leq m, n \leq 100)$. Chuyển vị ma trận (transpose) là hoán vị các dòng và các cột. Chúng ta đặt ký hiệu T lên góc phía trên bên phải của ma trận với nghĩa là chuyển vị.

$$\begin{pmatrix} 5 & 6 & 7 \\ 8 & 9 & 10 \end{pmatrix}^T = \begin{pmatrix} 5 & 8 \\ 6 & 9 \\ 7 & 10 \end{pmatrix}$$

Viết chương trình tạo ma trận chuyển vị của ma trân a

Input

- Dòng số đầu tiên chưa hai số nguyên: m, n
- m dòng tiếp theo, mỗi dòng chứa n số nguyên của bảng a

Output

- Gồm n dòng, mỗi dòng chứa m số nguyên là ma trận chuyển vị

Ví dụ

Input	Output
2 3 5 6 7 8 9 10	5 8 6 9 7 10

Bài 7. Khoảng cách Euclid giữa 2 ma trận

Cho hai ma trận số nguyên $a[m \times n], b[m \times n]$ với $(1 \leq m, n \leq 100)$. Khoảng cách Euclid của ma trận a và ma trận b được định nghĩa như sau:

$$dist(a, b) = \sqrt{\sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (a_{i,j} - b_{i,j})^2}$$

Ví dụ 1

$$a = \begin{pmatrix} 4 & 5 \\ 6 & 7 \end{pmatrix}$$

$$b = \begin{pmatrix} 2 & 5 \\ 8 & 1 \end{pmatrix}$$

$$dist(a,b) = \sqrt{(4-2)^2 + (5-5)^2 + (6-8)^2 + (7-1)^2} = \sqrt{4+0+4+36} = \sqrt{44} = 6.6332$$

Ví dụ 2

0	0	0	0	0	1	1	0	0	0
0	0	0	0	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	1	1	0	1	1	0	0	0
0	0	0	0	0	1	1	0	0	0
0	0	0	0	0	1	1	0	0	0
0	0	0	0	0	1	1	0	0	0
0	0	0	0	0	1	1	0	0	0
0	0	0	0	0	1	1	0	0	0
0	0	0	0	0	1	1	0	0	0

Ma trận a

0	0	0	0	0	1	1	1	0	0
0	0	0	0	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	1	1	1	0	1	1	0	0	0
0	0	1	0	0	1	1	0	0	0
0	0	0	0	0	1	0	0	0	0
0	0	0	0	1	1	1	0	0	0
0	0	0	0	0	1	1	0	0	0
0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	1	1	0	0	0

Ma trận b

Ta có khoảng cách giữa hai ma trận là

$$dist(a,b) = \sqrt{6} = 2.4495$$

Input

- Dòng số đầu tiên chưa hai số nguyên: m, n
- m dòng tiếp theo, mỗi dòng chứa n số nguyên của ma trận a
- m dòng tiếp theo, mỗi dòng chứa n số nguyên của ma trận b

Output

- Số thực là khoảng cách của 2 ma trận (**lấy 2 số lẻ**)

Ví dụ

Input	Output
2 2 4 5 6 7 2 5 8 1	6.63

Bài 8. Dot product của hai ma trận

Cho hai ma trận số nguyên $a[n \times n]$ và $b[n \times n]$ với $(1 \leq n \leq 100)$. Phép tích chấm (dot product) của ma trận a với ma trận b là một giá trị được tính như sau

$$value = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} a_{i,j} \times b_{i,j}$$

Ví dụ

Ma trận a

$$a = \begin{pmatrix} 2 & 1 & 2 \\ 1 & 1 & 1 \\ 4 & 5 & 6 \end{pmatrix}$$

Ma trận b

$$b = \begin{pmatrix} 4 & 5 & 6 \\ 2 & 3 & 4 \\ 1 & 2 & 3 \end{pmatrix}$$

Mảng kết quả phép nhân c

$$value = 2 \times 4 + 1 \times 5 + 2 \times 6 + 1 \times 2 + 1 \times 3 + 1 \times 4 + 4 \times 1 + 5 \times 2 + 6 \times 3 = 66$$

Input

- Dòng số đầu tiên chưa số nguyên: n
- n dòng tiếp theo, mỗi dòng chứa n số nguyên của ma trận a
- n dòng tiếp theo, mỗi dòng chứa n số nguyên của ma trận b

Output

- Một số là giá trị tích chấm của a và b

Ví dụ

Input	Output
3 2 1 2 1 1 1 4 5 6 4 5 6 2 3 4 1 2 3	66

Bài 9. Phép tính Convolution

Cho ma trận số nguyên $a[m \times n]$ và $b[k \times k]$ với ($1 \leq m, n, k \leq 100$ và $k < n;\ k < m$). Ta gọi ma trận a là ma trận lớn (a còn gọi là image), ma trận b là ma trận nhỏ (b còn gọi là kernel).

Phép tích chập (convolution) của ma trận nhỏ b lên ma trận lớn a được tính bằng cách: trượt ma trận nhỏ b lên ma trận lớn a từ trên xuống dưới, từ trái sang phải (ma trận b phải nằm gọn trong ma trận a). Tại mỗi vị trí trượt chúng ta tính tích chấm giữa ma trận b với vùng của ma trận a mà b đang được đặt lên trên.

Ví dụ

Ma trận a

$$a = \begin{pmatrix} 4 & 2 & 2 & 4 \\ 1 & 9 & 5 & 3 \\ 1 & 4 & 2 & 4 \\ 0 & 9 & 8 & 1 \end{pmatrix}$$

Ma trận b

$$b = \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

Mảng kết quả phép tích chập c

$$c = \begin{pmatrix} 51 & 10 \\ -3 & -27 \end{pmatrix}$$

Input

- Dòng số đầu tiên chưa số nguyên: m, n, k
- m dòng tiếp theo, mỗi dòng chứa n số nguyên của ma trận a
- k dòng tiếp theo, mỗi dòng chứa k số nguyên của ma trận b

Output

- Chứa $(m - k + 1)$ dòng, mỗi dòng chưa $(n - k + 1)$ giá trị của ma trận c

Ví dụ

Input	Output
4 4 3 4 2 2 4 1 9 5 3 1 4 2 4 0 9 8 1 −1 −1 −1 −1 8 −1 −1 −1 −1	51 10 −3 −27

Buổi 2. Kỹ thuật giá trị lính canh và đặt biến cờ

Sau khi hoàn thành bài thực hành này sinh viên có thể:

- Sử dụng kỹ thuật giá trị lính canh
- Sử dụng kỹ thuật đặt biến cờ

Bài 1. Điểm trung bình

Trong một lớp học, các sinh viên đều phải làm một bài kiểm tra. Điểm của các bài kiểm tra có giá trị từ 0 đến 10. Hãy xác định điểm trung bình của bài kiểm tra này

Input

- Điểm của một sinh viên nằm trên một dòng
- Dòng cuối cùng là số -1 báo hiệu hết dữ liệu

Output

- Dòng thứ nhất chứa số lượng sinh viên
- Dòng thứ hai chứa điểm trung bình của lớp (lấy 2 số lẻ thập phân)

Ví dụ

Input	Output
4 10 8 9 9 -1	5 8.00

Hướng dẫn: Dùng kỹ thuật "giá trị lính canh"

Bài 2. Kiểm tra giá trị trong mảng

Cho mảng số nguyên $\boldsymbol{a} = (a_0, a_1, \dots, a_{n-1}), (1 \le n \le 10^8)$ và số nguyên $\boldsymbol{x}$. Kiểm tra xem mảng $\boldsymbol{a}$ có chứa giá trị $\boldsymbol{x}$ không? Nếu có xuất ra "Yes", ngược lại xuất ra "No"

Input

- Dòng đầu chứa số nguyên n và x
- Dòng thứ hai chứa các số nguyên mảng a

Output

- Dòng thứ nhất chứa số lượng sinh viên
- Dòng thứ hai chứa điểm trung bình của lớp (lấy 2 số lẻ thập phân)

Ví dụ

Input	Output
6 4 8 8 9 9 4 2	Yes

Input	Output
6 12 8 8 9 9 4 2	No

Hướng dẫn:

- Dùng kỹ thuật "giá trị lính canh" hay
- Dùng kỹ thuật "đặt biến cờ"

Bài 3. Kiểm tra số nguyên tố

Cho số nguyên n ($1 \leq n \leq 10^6$). Viết hàm kiểm tra số n có là số nguyên tố hay không. Nếu là số nguyên tố xuất ra "Yes", ngược lại xuất ra "No"

Input

- Dòng duy nhất chứa số nguyên n

Output

- Dòng duy nhất chứa "Yes" hay "No"

Ví dụ

Input	Output
9	No

Input	Output
13	Yes

Hướng dẫn: Dùng kỹ thuật "đặt biến cờ"

Buổi 3. Kỹ thuật mảng đánh dấu và mảng đếm

Sau khi hoàn thành bài thực hành này sinh viên có thể:

- Sử dụng kỹ thuật mảng đánh dấu trạng thái
- Sử dụng kỹ thuật mảng đếm

Bài 1. Số nhỏ nhất

Cho n số nguyên dương $a = (a_1, a_2, \dots, a_n)$ ($1 \le n \le 10^8$ và $1 \le a_i \le 10^6$). Hãy tìm số nguyên dương nhỏ nhất không xuất hiện trong a.

Input

- Dòng đầu tiên chứa số nguyên n
- Dòng thứ hai chứa n số tự nhiên

Output

- Số tự nhiên nhỏ nhất không xuất hiện trong a

Ví dụ

Input	Output
8 6 7 1 2 5 3 2 6	4

Hướng dẫn: Dùng kỹ thuật "mảng đánh dấu trạng thái"

Bài 2. Sàng nguyên tố

Cho số nguyên n ($1 \le n \le 10^6$). Viết chương trình liệt kê các số nguyên tố nhỏ hơn hay bằng n.

Ví dụ 1

- $n = 10$
- Các số nguyên tố nhỏ hơn 10: $2, 3, 5, 7$

Ví dụ 2

- $n = 20$
- Các số nguyên tố nhỏ hơn 20: $2, 3, 5, 7, 11, 13, 17, 19$

Input

- Dòng duy nhất chứa số nguyên n

Output

- Dòng thứ nhất chứa số m là số lượng số nguyên tìm được
- Dòng thứ hai chứa m số nguyên tố nhỏ hơn n

Ví dụ

Input	Output
10	4 2 3 5 7

Input	Output
20	8 2 3 5 7 11 13 17 19

Hướng dẫn: Dùng kỹ thuật "mảng đánh dấu trạng thái"

Thuật toán "Eratosthene"

1. **Cấu trúc dữ liệu:** dùng một mảng a để đánh dấu số nào là số nguyên tố, số nào không phải là số nguyên tố.

$$a[i] = \begin{cases} true & nếu\ i\ là\ số\ nguyên\ tố \\ false & nếu\ i\ không\ phải\ là\ số\ nguyên\ tố \end{cases}$$

2. **Ý tưởng:**
   - Ban đầu, chúng ta có tập các số $\{2,3,\dots,n\}$
   - Tại mỗi bước, chúng ta chọn số nhỏ nhất trong tập (số nhỏ nhất này là số nguyên tố) và bỏ đi các bội của số đó
3. **Cải tiến**
   - Mọi số không nguyên tố có ước số $\leq \sqrt{n}$ → chúng ta chỉ cần bỏ các bội của **số nguyên tố $\leq \sqrt{n}$**

Bài 3. Tìm các tổng

Cho n gói kẹo được đánh số từ 1 đến n. Số lượng kẹo trong các gói được cho trong mảng $a = (a_1, a_2, \dots, a_n)$ ($1 \leq n \leq 200$ và $0 \leq a_i \leq 1000$). Một số gói kẹo có thể gôm lại thành một phần và số kẹo trong một phần là tổng số kẹo trong của các gói trong phần đó. Hãy cho biết các loại tổng khác nhau của các phần

Ví dụ: $a = (2,5,4)$. Ta có 7 phần có tổng khác nhau là

- $2 = a_1$
- $4 = a_3$
- $5 = a_2$
- $6 = a_1 + a_3$
- $7 = a_1 + a_2$
- $9 = a_2 + a_3$
- $11 = a_1 + a_2 + a_3$

Input

- Dòng đầu tiên chứa số nguyên n
- Dòng thứ hai chứa n số

Output

- Dòng đầu tiên chứa số lượng tổng khác nhau
- Dòng thứ hai chứa các giá trị tổng (từ nhỏ đến lớn)

Ví dụ

Input	Output
3 2 5 4	7 2 4 5 6 7 9 11

Hướng dẫn: Dùng kỹ thuật "mảng đánh dấu trạng thái"

1. **Cấu trúc dữ liệu:** dùng một mảng $tong$ để đánh dấu tổng nào có thể được tạo ra từ dãy số.

$$tong[i] = \begin{cases} true & nếu\ i\ là\ tổng\ được\ sinh\ ra \\ false & nếu\ i\ là\ không\ tổng\ được\ sinh\ ra \end{cases}$$

2. **Ý tưởng:**
   - Xét từng số $a[i]$
   - Với số $a[i]$, xét các tổng k đã được sinh ra (xét k từ lớn đến nhỏ) nếu $tong[k + a[i]] = false$ thì $tong[k + a[i]] = true$

Bài 4. Đếm số lượng mỗi số dương

Cho n số nguyên dương $a = (a_1, a_2, \dots, a_n)$ ($1 \le n \le 10^8$ và $0 \le a_i \le 10^6$). Hãy cho biết mỗi số nguyên trong dãy a xuất hiện bao nhiêu lần

Ví dụ: $a = (2,2,5,5,2,3,5,4,2,4)$

- Số **2**: xuất hiện **4 lần**
- Số **3**: xuất hiện **1 lần**
- Số **4**: xuất hiện **2 lần**
- Số **5**: xuất hiện **3 lần**

Input

- Dòng đầu tiên chứa số nguyên n
- Dòng thứ hai chứa n số nguyên

Output

- Mỗi dòng xuất ra theo định dạng ***i*: *k*** với ý nghĩa là số i xuất hiện k lần (các số i được xuất theo thứ tự từ nhỏ đến lớn)

Ví dụ

Input	Output
10 2 2 5 5 2 3 5 4 2 4	2:4 3:1 4:2 5:3

Hướng dẫn: Dùng kỹ thuật "mảng đếm"

Bài 5. Đếm số lượng mỗi số

Giải quyết bài toán trên cho trường hợp tổng quá hơn: dãy a chứa vừa số nguyên âm, vừa chứa số nguyên dương ($-10^6 \leq a_i \leq 10^6$)

Buổi 4. Kỹ thuật sắp xếp và tìm kiếm

Sau khi hoàn thành bài thực hành này sinh viên có thể:

- Sử dụng kỹ thuật sắp xếp: Interchange sort, quick sort, hàm Array.Sort
- Sử dụng kỹ thuật tìm kiếm: tuyến tính, nhị phân

Bài 1. Sắp xếp

Cho n số nguyên $a = (a_1, a_2, \dots, a_n)$ ($\mathbf{1 \le n \le 5000}$). Hãy sắp xếp dãy a tăng dần theo thuật toán Interchange sort hay Selection sort

Input

- Dòng đầu tiên chứa số nguyên n
- Dòng thứ hai chứa n số nguyên

Output

- Dòng duy nhất các số nguyên đã sắp xếp

Ví dụ

Input	Output
8 6 5 3 7 8 2 12 14	2 3 5 6 7 8 12 14

Bài 2. Sắp xếp nhanh

Cho n số nguyên $a = (a_1, a_2, \dots, a_n)$ ($\mathbf{1 \le n \le 10^6}$). Hãy sắp xếp dãy a tăng dần theo thuật toán **quick sort**

Input

- Dòng đầu tiên chứa số nguyên n
- Dòng thứ hai chứa n số nguyên

Output

- Dòng duy nhất các số nguyên đã sắp xếp

Ví dụ

Input	Output
8 6 5 3 7 8 2 12 14	2 3 5 6 7 8 12 14

Hướng dẫn:

- Gọi hàm **Array.Sort** hay
- Tự cài đặt thuật toán QuickSort

Bài 3. Couting sort

Cho n số nguyên $a = (a_1, a_2, \dots, a_n)$ ($\mathbf{1 \le n \le 10^8, 0 \le a_i \le 10^6}$). Hãy sắp xếp dãy a tăng dần

Input

- Dòng đầu tiên chứa số nguyên n
- Dòng thứ hai chứa n số nguyên

Output

- Dòng duy nhất các số nguyên đã sắp xếp

Ví dụ

Input	Output
8 6 8 2 5 8 2 12 13	2 2 5 6 8 8 12 13

Hướng dẫn: Dùng kỹ thuật "mảng đếm"

Bài 4. Số nhỏ nhất

Cho n số nguyên $a = (a_1, a_2, \dots, a_n)$ ($1 \le n \le 10^6$ và $1 \le a_i \le 10^9$). Hãy tìm số nguyên nhỏ nhất không xuất hiện trong a.

Input

- Dòng đầu tiên chứa số nguyên n
- Dòng thứ hai chứa n số nguyên dương

Output

- Số nguyên dương nhỏ nhất không xuất hiện trong a

Ví dụ

Input	Output
8 6 7 1 2 5 3 2 6	4

Hướng dẫn:

- Sắp xếp dãy a tăng dần
- Dùng kỹ thuật lính canh

Bài 5. Đếm số khác nhau

Cho n số nguyên $a = (a_1, a_2, \dots, a_n)$ $(1 \le n \le 10^6)$. Hãy đếm số lượng các số khác nhau trong dãy a.

Input

- Dòng đầu tiên chứa số nguyên n
- Dòng thứ hai chứa n số nguyên

Output

- Dòng duy nhất chứa số nguyên là số lượng số khác nhau trong dãy a

Ví dụ

Input	Output
8 6 6 1 2 5 3 2 6	5

Hướng dẫn:

- Sắp xếp dãy a tăng dần

Bài 6. Tìm kiếm nhị phân

Cho n số nguyên $a = (a_1, a_2, \dots, a_n)$ $(1 \le n \le 10^6)$ thỏa điều kiện $a_1 \le a_2 \le \cdots \le a_n$ và số nguyên x. Hãy tìm ví trí xuất hiện của x trong a

Input

- Dòng đầu tiên chứa số nguyên n và số x
- Dòng thứ hai chứa n số nguyên

Output

- Xuất ra -1 nếu x không có trong a, ngược lại xuất ra vị trí của x trong a

Ví dụ

Input	Output
10 12 3 6 10 12 16 20 22 23 26 28	4

Bài 7. Tìm cặp số

Cho n số nguyên $a = (a_1, a_2, \dots, a_n)$ ($1 \le n \le 10^6$). Hãy cho biết trong a có hai chỉ số $i < j$ sao cho $a_i + a_j = 0$ không?

Input

- Dòng đầu tiên chứa số nguyên n
- Dòng thứ hai chứa n số nguyên

Output

- Nếu trong a có cặp thỏa yêu cầu bài toán thì in "Yes", ngược lại in "No"

Ví dụ

Input	Output
8 3 2 -5 2 5 3 -7 1	Yes

Hướng dẫn:

- Sắp xếp dãy a tăng dần
- Xét mỗi vị trị i, dùng tìm kiếm nhị phân để tìm vị trí j

Buổi 5. Kỹ thuật xử lý chuỗi

Sau khi hoàn thành bài thực hành này sinh viên có thể:

- Sử dụng kỹ thuật xử lý chuỗi: xét từng ký tự, chuyển đổi ký tự sang số, sử dụng được một số hàm trên chuỗi, …
- Giải một số bài toán liên quan đến chuỗi

Bài 1. Ký tự

Cho chuỗi s (có độ dài nhỏ hơn 10^6), chỉ gồm các ký tự từ 'a' đến 'z' và khoảng trắng. Hãy cho biết có bao nhiêu loại ký tự xuất hiện trong s (không tính khoảng trắng). Và cho biết ký tự xuất hiện nhiều nhất (không tính khoảng trắng), nếu có nhiều ký tự khác nhau có cùng số lượng xuất hiện, chọn ký tự nhỏ nhất (theo thứ tự từ điển)

Input

- Dòng duy nhất chứa chuỗi s

Output

- Dòng đầu tiên chứa số lượng loại ký tự xuất hiện trong s
- Dòng thứ hai chứa ký tự nhỏ nhất xuất hiện nhiều nhất

Ví dụ

Input	Output
dai hoc ngoai ngu tin hoc tphcm	12 c

Hướng dẫn: Dùng kỹ thuật "mảng đếm"

Bài 2. Kiểm tra Password

Cho chuỗi s là password do người dùng nhập vào. Hãy kiểm tra password có thỏa các ràng buộc sau hay không

- Có ít nhất 8 ký tự
- Phải chứa ký tự thường, ký tự hoa và ký tự số

Input

- Gồm nhiều dòng, mỗi dòng chứa một password
- Dòng cuối cùng ghi −1 báo hiệu kết thúc

Output

- Gồm nhiều dòng, mỗi dòng ghi "Yes" nếu password ở dòng tương ứng hợp lệ, "No" nếu password không hợp lệ

Ví dụ

Input	Output
abc123456 123456789 Abcd12 Abcd123456 -1	No No No Yes

Bài 3. Tách họ tên

Cho chuỗi *s* chứa họ tên của một người. Hãy viết chương trình tách họ, tên, và tên lót trong chuỗi *s*

Input

- Dòng duy nhất chứa chuỗi *s*

Output

- Dòng đầu tiên chứa họ
- Dòng thứ hai chứa tên lót
- Dòng thứ ba chứa tên

Ví dụ

Input	Output
Nguyễn Ngọc Phương Trinh	Nguyễn Ngọc Phương Trinh

Bài 4. Chuỗi không dấu

Cho chuỗi *s* chứa câu tiếng Việt có dấu. Hãy chuyển chuỗi *s* sang tiếng Việt không dấu

Ví dụ:

Câu có dấu: Trường Đại học HUFLIT

Câu không dấu: Truong Dai hoc HUFLIT

Input

- Dòng duy nhất chứa chuỗi s

Output

- Dòng duy nhất chứa chuỗi không dấu

Ví dụ

Input	Output
Trường Đại học HUFLIT	Truong Dai hoc HUFLIT

Bài 5. Danh sách họ tên

Cho một danh sách chứa họ tên của n người $(1 \leq n \leq 1000)$. Hãy sắp xếp danh sách họ tên theo họ hay theo tên

Input

- Dòng thứ nhất chứa số nguyên n và số nguyên k
  - $k = 1$ sắp xếp theo họ
  - $k = 2$ sắp xếp theo tên
- n dòng sau, mỗi dòng chứa một họ tên của một người

Output

- n dòng, mỗi dòng là một họ tên đã được sắp xếp

Ví dụ

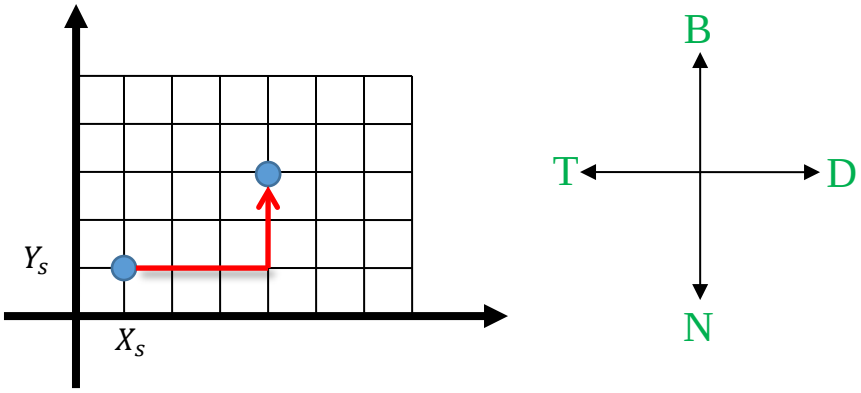
Input	Output
4 2 Lê Minh Hải Trần An Trần Thanh Nguyên Minh Hiếu	Trần **An** Lê Minh **Hải** Nguyên Minh **Hiếu** Trần **Thanh**

Hướng dẫn:

- Tách họ, tên
- Dùng kỹ thuật "Sắp xếp"

Bài 6. Robot đi theo hướng đông, tây, nam, bắc

Một Robot xuất phát từ điểm có tọa độ (X_S, Y_S) trong mặt phẳng tọa độ nguyên OXY, mỗi lần Robot đi được một đơn vị độ dài theo một trong 4 hướng: Đông (D), Tây (T), Nam (N), Bắc (B).



Yêu cầu: Cho một chương trình s dùng để điều khiển Robot, s gồm một dãy các ký tự D, T, N, B ($1 \leq length(s) \leq 10000$). Hãy tìm tọa độ của Robot sau khi Robot thực hiện xong chương trình.

Input

- Dòng thứ nhất chứa hai số nguyên X_S, Y_S
- Dòng thứ hai chứa chương trình s

Output

- Dòng duy nhất chứa hai số nguyên $X_{D,}Y_D$ là tọa độ của Robot sau khi thực hiện chương trình

Ví dụ

Input	Output
1 1 DDDBB	4 3

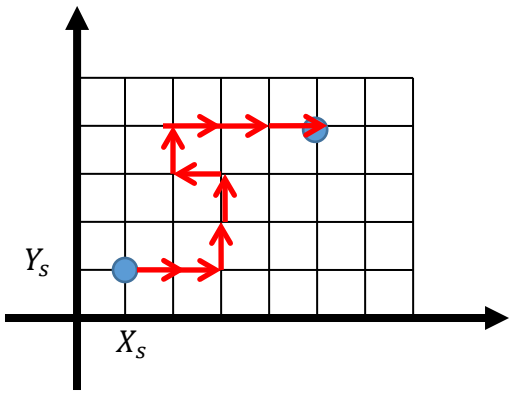
Bài 7. Robot có mắt

Một Robot được đặt tại điểm có tọa độ (X_S, Y_S) trong mặt phẳng tọa độ nguyên OXY và **mắt của Robot hướng sang phải**. Robot được điều khiển bởi một chương trình gồm một dãy các lệnh: F (Forward) , B (Backward), L (Left), R (Right). Trong đó:

- F: Đi về phía trước một đơn vị so với mắt của Robot
- B: Đi lui về phía sau một đơn vị so với mắt của Robot
- R: Quay sang phải một đơn vị so với mắt của Robot và mắt của Robot cũng thay đổi theo.
- L: Quay sang trái một đơn vị so với mắt của Robot và mắt của Robot cũng thay đổi theo.

Chú ý: Mỗi lần Robot rẽ theo hướng nào thì mắt của Robot đi theo hướng đó.

Ví dụ: Giả sử Robot được đặt tại $(X_S, Y_S) = (1,1)$ và $s = FFLFLRRFF$ thì Robot sẽ đi như hình vẽ sau:



Yêu cầu: Cho một chương trình s điều khiển Robot gồm một dãy các ký tự F, B, L, R ($1 \leq length(s) \leq 10000$). Hãy tìm tọa độ cuối cùng của Robot sau khi Robot thực hiện xong chương trình.

Input

- Dòng thứ nhất chứa hai số nguyên X_S, Y_S
- Dòng thứ hai chứa chương trình s

Output

- Dòng duy nhất chứa hai số nguyên $X_{D,}Y_D$ là tọa độ của Robot sau khi thực hiện chương trình

Ví dụ

Input	Output
1 1 FFLFLRRFF	5 4

Bài 8. Cộng hai số vô cùng lớn

Để tính toán (cộng, trừ, nhân, chia) các số có hàng ngàn, hàng triệu chữ số chúng ta không thể sử dụng các kiểu dữ liệu int, double. Một giải pháp khả thi là chúng ta lưu trữ các số lớn vào chuỗi, mỗi số là một chuỗi, sau đó viết thuật toán mô phỏng cách con người thực hiện các phép toán trên các chuỗi.

Yêu cầu: cho hai số nguyên dương lớn được lưu trong hai chuỗi s_1, s_2. Hãy viết chương trình tính tổng hai số s_1, s_2

Input

- Dòng đầu tiên chứa chuỗi s_1

- Dòng thứ hai chứa chuỗi s_2

Output

- Dòng duy nhất chứa kết quả là tổng của hai số

Ví dụ

Input	Output
123456 876	124332

Hướng dẫn:

- Bước 1. Thêm các số 0 vào đầu chuỗi của chuỗi có độ dài ngắn hơn sao cho 2 chuỗi có độ dài bằng nhau

```
123456          123456
876      ==>    000876
------          ------
```

- Bước 2. Thực hiện phép cộng từng chữ số từ phải sang trái, chú ý khi cộng có thể phát sinh ra giá trị nhớ cho số kế bên.

  Ví dụ:

  - $6 + 6 = 12$ viết 2 nhớ 1.
  - $5 + 7 = 12$, công thêm nhớ 1 trước đó và được kết quả là 13. Viết 3 nhớ 1

```
123456
000876      Nhớ = 1
------
     2
```

- Bước 3. Sau khi cộng nếu còn nhớ thì phải thêm nhớ vào đầu chuỗi kết quả

Buổi 6. Kỹ thuật đệ quy

Sau khi hoàn thành bài thực hành này sinh viên có thể: Sử dụng kỹ thuật đệ quy

- Cài đặt chương trình theo công thức đệ quy
- Sử dụng phương pháp chia để trị

Bài 1. Tính giai thừa

Cho số nguyên n $(0 \leq n \leq 12)$. Viết hàm đệ quy tính $n!$ theo công thức đệ quy sau

$$n! = \begin{cases} 1, & nếu\ x \leq 1 \\ n \times (n-1)!, & nếu\ x > 1 \end{cases}$$

Input

- Dòng duy nhất chứa số nguyên n

Output

- Giá trị giai thừa tính được

Ví dụ

Input	Output
6	720

Bài 2. Tính hàm số mũ

Cho hai số nguyên a, n $(a > 0, n \geq 0)$. Viết hàm đệ quy tính a^n theo công thức đệ quy sau

$$a^n = \begin{cases} 1, & nếu\ n = 0 \\ a^{\frac{n}{2}} \times a^{\frac{n}{2}}, & nếu\ n\ chẵn \\ a \times a^{\frac{n-1}{2}} \times a^{\frac{n-1}{2}}, & nếu\ n\ lẻ \end{cases}$$

Input

- Dòng duy nhất chứa hai số nguyên a và n

Output

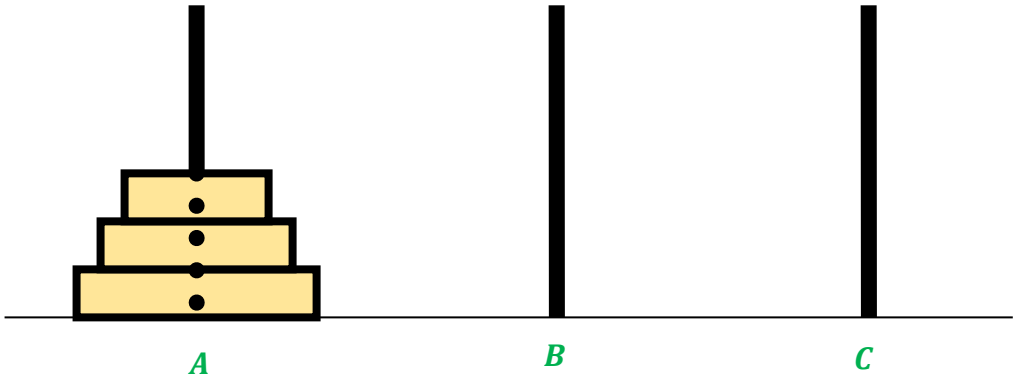
- Giá trị a^n tính được

Ví dụ

Input	Output
2 6	64

Bài 3. Tháp Ha Noi

Có 3 cột được đặt tên là A, B, C. Cột A chứa n cái đĩa và các đĩa nhỏ được đặt trên đĩa lớn (xem hình phía dưới).



Ba cột và cột A chứa $n = 3$ đĩa

Hãy thực hiện từng bước để chuyển n đĩa từ cột A sang cột C với các ràng buộc sau:

- Mỗi lần chỉ được di chuyển một đĩa (đĩa nằm trên cùng của cột) từ cột này sang cột khác (có thể từ cột A sang B hay sang C; từ cột B sang A hay sang C; từ cột C sang A hay sang B)
- Sau mỗi bước chuyển, ở các cột, đĩa lớn luôn nằm dưới, đĩa nhỏ nằm trên

Input

- Dòng duy nhất chứa số nguyên n

Output

- Gồm nhiều dòng, mô tả các bước di chuyển đĩa, theo định dạng:

Chuyển 1 đĩa từ cột X sang cột Y

Trong đó $X, Y \in \{A, B, C\}$

Ví dụ 1

Input	Output
2	Chuyển 1 đĩa từ cột A sang cột B Chuyển 1 đĩa từ cột A sang cột C Chuyển 1 đĩa từ cột B sang cột C

Ví dụ 2

Input	Output
3	Chuyển 1 đĩa từ cột A sang cột C Chuyển 1 đĩa từ cột A sang cột B Chuyển 1 đĩa từ cột C sang cột B Chuyển 1 đĩa từ cột A sang cột C Chuyển 1 đĩa từ cột B sang cột A Chuyển 1 đĩa từ cột B sang cột C Chuyển 1 đĩa từ cột A sang cột C

Hướng dẫn: Sử dụng phương pháp chia để trị

- Chuyển $(n-1)$ đĩa từ cột A sang cột B, lấy cột C làm cột trung gian
- Chuyển 1 đĩa từ cột A sang cột C
- Chuyển $(n-1)$ đĩa từ cột B sang cột C, lấy cột A làm cột trung gian

Bài 4. Đọc số

Cho số nguyên dương n $(0 \leq n \leq 2 \times 10^9)$. Hãy chuyển số n thành câu tiếng Việt.

Ví dụ:

123: một trăm hai mươi ba

104: một trăm lẻ bốn

1234: một ngàn hai trăm ba mươi bốn

Input

- Dòng duy nhất chứa số nguyên n

Output

- Dòng duy nhất chứa chuỗi kết quả

Ví dụ

Input	Output
123	một trăm hai mươi ba

Hướng dẫn: sử dụng kỹ thuật đệ quy và phương pháp chia để trị

Buổi 7. Kỹ thuật Đóng gói

Sau khi hoàn thành bài thực hành này sinh viên có thể: Sử dụng kỹ thuật hướng đối tượng theo nguyên tắc đóng gói:

- Tạo lớp
- Tạo đối tượng
- Tạo danh sách đối tượng

Bài 1. Phân số

Xây dựng lớp **PhanSo** để biểu diễn phân số với các thuộc tính **TuSo** và **MauSo** nguyên (chỉ truy cập được bên trong lớp). Định nghĩa các phương thức sau:

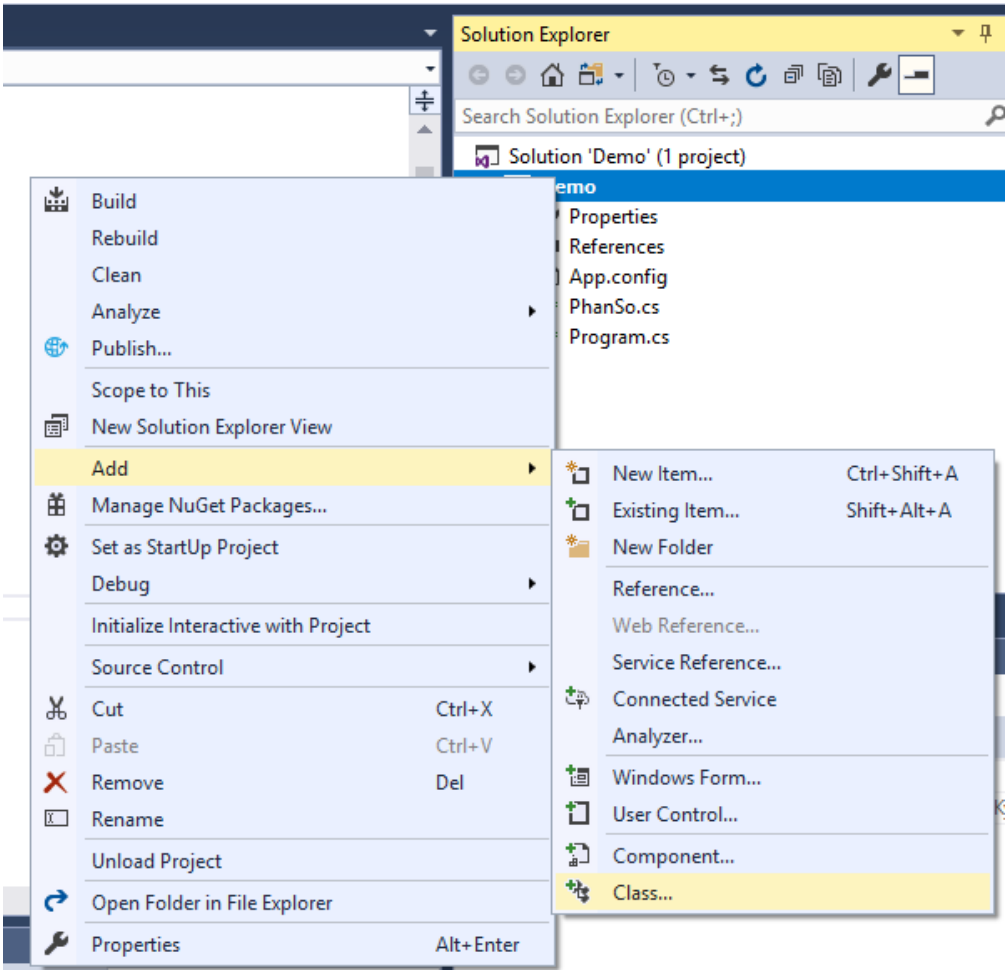
- Khởi tạo phân số (phân số được rút gọn sau khi khởi tạo).
- Nhập giá trị phân số từ bàn phím.
- In giá trị phân số ra màn hình.
- Tính giá trị thập phân của phân số.
- Cộng 2 phân số.
- Trừ 2 phân số.
- Nhân 2 phân số.
- Chia 2 phân số.

Viết chương trình nhập vào 2 phân số. In ra màn hình:

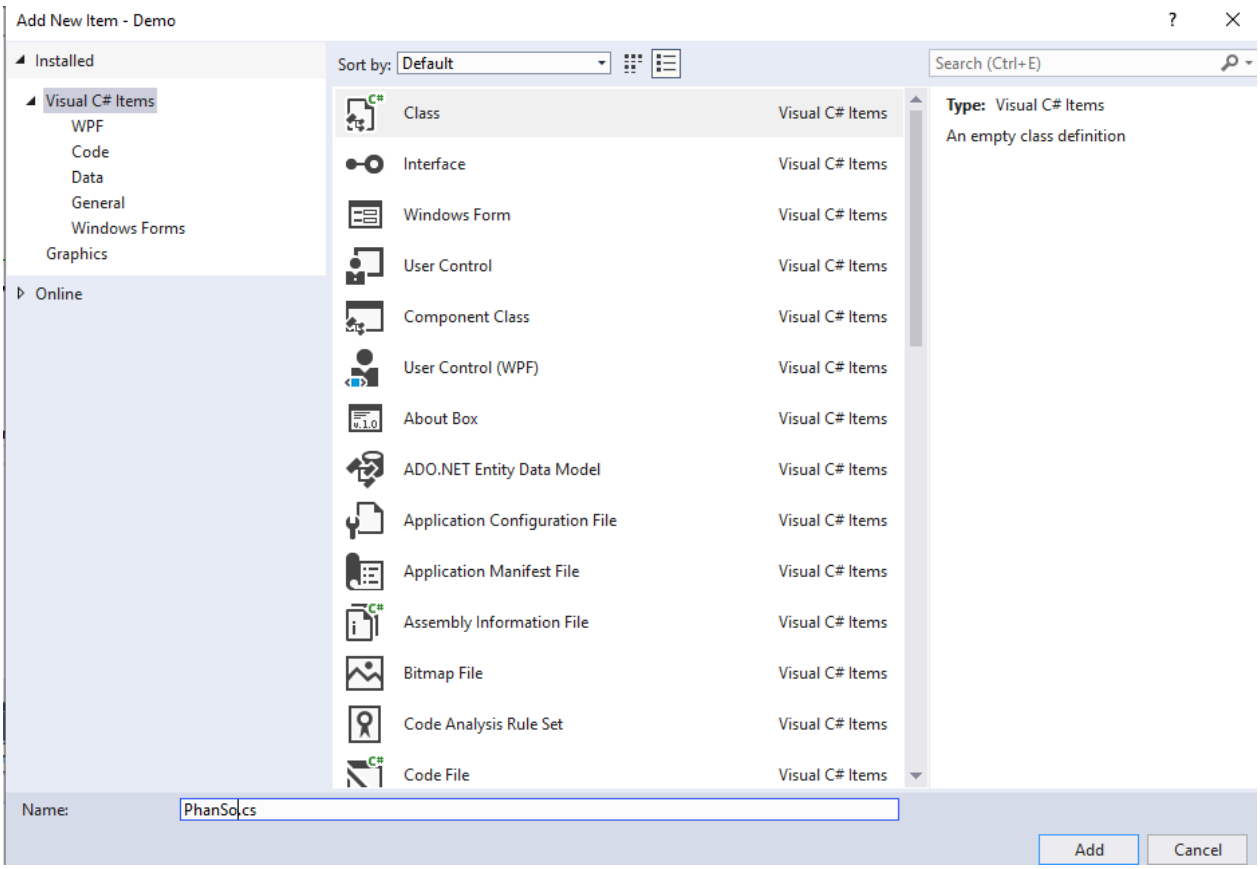
- Giá trị 2 phân số sau khi đã rút gọn.
- Giá trị thập phân của 2 phân số.
- Kết quả cộng, trừ, nhân, chia 2 phân số.

Hướng dẫn:

- **Bước 1:** Tạo Console App (.NET Framework) trên Visual Studio.
- **Bước 2:** Tạo **Class** mới bằng cách click chuột phải vào tên **Project → Add → Class**.



- **Bước 3:** Đặt tên cho Class mới là **PhanSo** → **Add**.



- **Bước 4:** Định nghĩa lớp **PhanSo** theo cấu trúc sau:

```
class PhanSo
{
    private int tuSo;
    private int mauSo;

    //Ham tinh Uoc so chung lon nhat cua 2 so nguyen a, b
    private int GCD(int a, int b)
    {
        ...
    }

    public PhanSo(int ts, int ms)
    {
        ...
    }

    public void Nhap()
    {
        ...
    }

    public void In()
    {
        ...
    }

    public double GiaTriThapPhan()
    {
        ...
    }

    public PhanSo Cong(PhanSo p)
    {
        ...
    }

    public PhanSo Tru(PhanSo p)
    {
        ...
    }

    public PhanSo Nhan(PhanSo p)
    {
        ...
    }

    public PhanSo Chia(PhanSo p)
    {
        ...
    }
}
```

- **Bước 5:** Quay trở lại hàm **Main(…)**. Khai báo 2 biến kiểu **PhanSo** và thực hiện các thao tác tính toán trên 2 đối tượng kiểu **PhanSo** vừa tạo.

Bài 2. Point2D

Xây dựng lớp **Point2D** biểu diễn các điểm trong không gian 2 chiều với các tọa độ nguyên.

Lớp **Point2D** cung cấp các thao tác thông dụng trên điểm như sau:

- Các constructor khởi tạo:
  - `Point2D()`: khởi tạo điểm ở gốc tọa độ (0,0).
  - `Point2D(int x, int y)`: khởi tạo điểm có hoành độ x, tung độ y.
  - `Point2D(Point2D p)`: khởi tạo điểm ở vị trí của điểm p.
- Nhập tọa độ cho điểm từ bàn phím: `void Input()`.
- Trả về chuỗi biểu diễn tọa độ của điểm dưới dạng "(x, y)": `string ToString()`.
- Di chuyển điểm đến tọa độ mới: `void Move(int x, int y)`.
- Kiểm tra xem điểm có phải là gốc tọa độ (0,0) hay không: `bool isOrigin()`.
- Hàm tính khoảng cách từ điểm tới điểm p: `double Distance(Point2D p)`.
- Hàm tĩnh tính khoảng cách giữa 2 điểm p1 và p2: `static double Distance(Point2D p1, Point2D p2)`.

Bài 3. Triangle

Xây dựng lớp **Triangle** biểu diễn các thông tin của một tam giác trong không gian 2 chiều với 3 đỉnh kiểu **Point2D**.

Lớp **Triangle** cung cấp các thao tác thông dụng trên điểm như sau:

- Constructor khởi tạo:
  - `Triangle(Point2D p1, Point2D p2, Point2D p3)`
- Tính chu vi của tam giác: `void Perimeter(int x, int y)`.
- Tính diện tích của tam giác: `double Area()`.

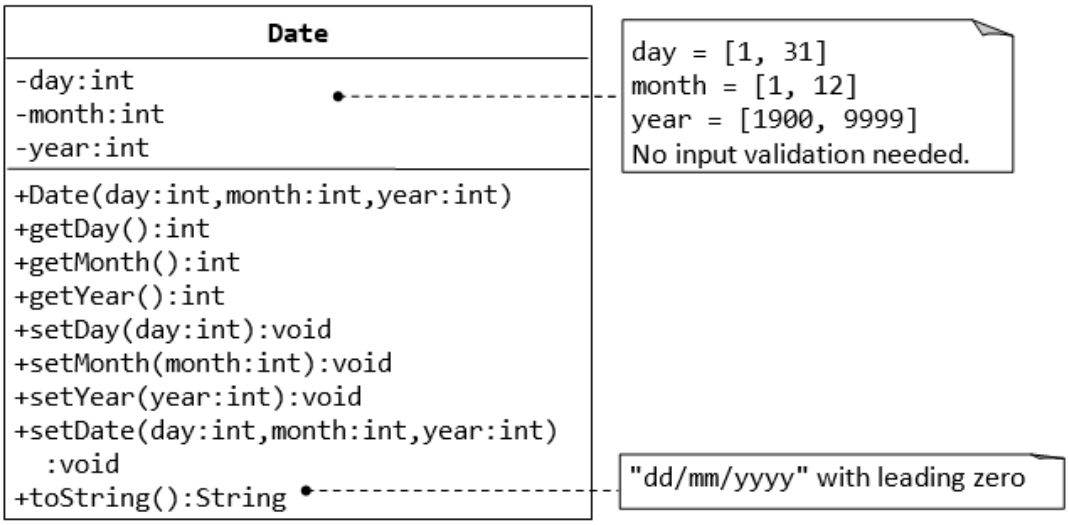
Bài 4. Rectangle

Xây dựng lớp **Rectangle** để biểu diễn hình chữ nhật với các thuộc tính mô tả chiều dài và chiều rộng. Định nghĩa các phương thức sau:

- Các constructor khởi tạo có tham số và không có tham số.
- Nhập thông tin hình chữ nhật.
- Xuất thông tin hình chữ nhật: "Rectangle[length=?, width=?]"
- Tính chu vi hình chữ nhật.
- Tính diện tích hình chữ nhật.
- Tính đường chéo hình chữ nhật.

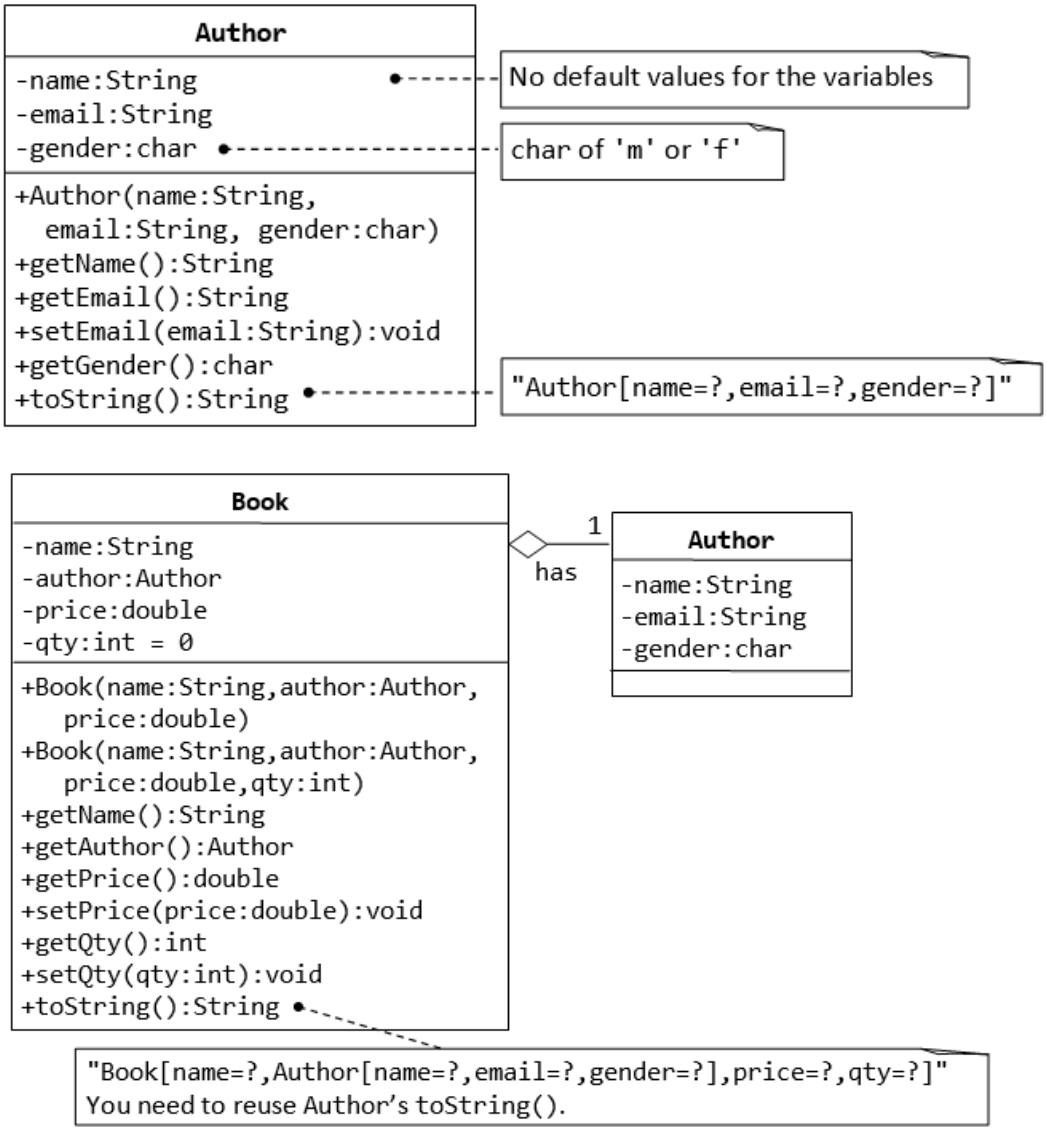
Bài 5. Date

Xây dựng lớp **Date** theo bản thiết kế sau:



Bài 6. Book

Xây dựng lớp **Author** và **Book** theo bản thiết kế sau:

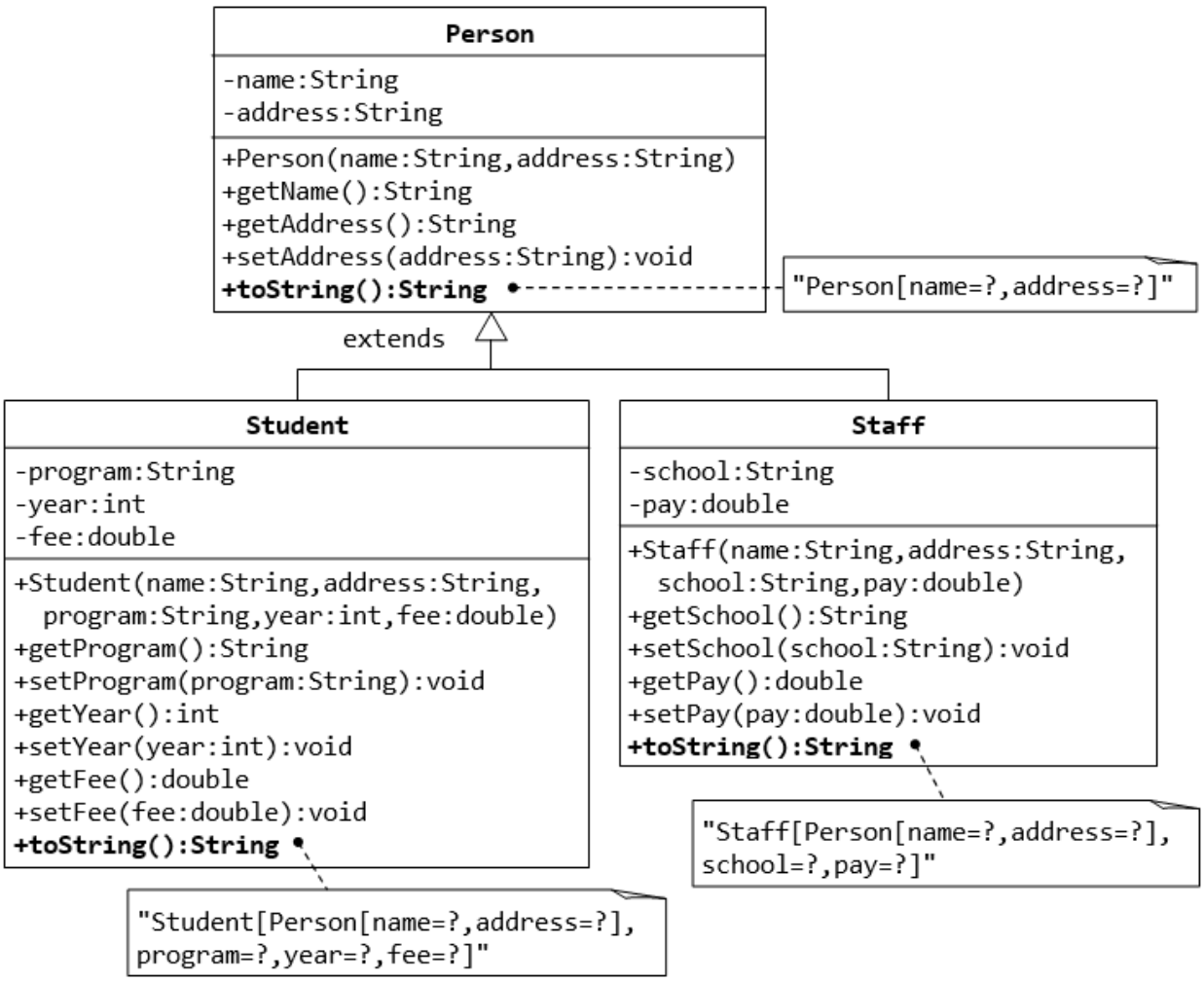


Buổi 8. Kỹ thuật Thừa kế và Đa hình

Sau khi hoàn thành bài thực hành này sinh viên có thể: Sử dụng kỹ thuật hướng đối tượng theo nguyên tắc thừa kế:

- Tạo lớp cha
- Tạo lớp con
- Overriding các methods

Bài 1. Xây dựng lớp **Person** và 2 lớp **Student**, **Staff** kế thừa từ lớp **Person** như bản thiết kế sau:



Hướng dẫn:

- **Bước 1:** Tạo lớp cơ sở **Person**.

```
class Person
{
	...
	public virtual void toString()
	{
		...
	}

}
```

- **Bước 2:** Tạo lớp **Student** kế thừa từ lớp **Person**.
  - Tạo một lớp mới đặt tên là **Student**.
  - Khai báo lớp **Student** kế thừa từ lớp **Person**.

```
class Student : Person
{
	...
}
```

  - Khai báo các thuộc tính riêng của lớp **Student**.
  - Xây dựng các phương thức cho lớp **Student**.
  - Overriding phương thức `toString()`.

```
class Student : Person
{
	...
	public override void toString()
	{
		...
	}

}
```

- **Bước 3:** Tạo lớp **Staff** kế thừa từ lớp **Person** tương tự như lớp **Student**.

Bài 2. Xây dựng chương trình theo sơ đồ lớp dưới đây:

Buổi 9. Kỹ thuật Operator overloading

Sau khi hoàn thành bài thực hành này sinh viên có thể: Sử dụng kỹ thuật operator overloading

Bài 1. Số phức

Xây dựng lớp **Complex** biểu diễn khái niệm số phức với hai thành phần dữ liệu là phần thực và phần ảo.

- Định nghĩa các phương thức xuất, nhập, cộng, trừ, nhân, chia hai số phức.
- Overriding các operator +, -, *, / trên lớp số phức.

Ví dụ: Cho hai số phức $A(a_1, a_2)$, $B(b_1, b_2)$

- $A + B = (a_1+b_1, a_2+b_2)$
- $A - B = (a_1-b_1, a_2-b_2)$
- $A * B = (a_1*b_1 - a_2*b_2, a_1*b_2+a_2*b_1)$
- $A / B = \left(\frac{a_1 * b_1 + a_2 * b_2}{b_1^2 + b_2^2}, \frac{b_1 * a_2 - a_1 * b_2}{b_1^2 + b_2^2} \right)$

Viết chương trình cho phép nhập vào hai số phức, in ra kết quả các phép toán cộng, trừ, nhân, chia hai số phức kể trên.

Bài 2. Phân số

Overloading các operator +, -, *, /, ==, !=, >, <, <=, >= của Class **PhanSo** đã xây dựng ở Bài tập 1 Buổi 7.