MÔN : LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG
Bài thực hành số 8.3 : Xây dựng chương trình tính tổng 2 ma trận

I. Mục tiêu :

- Giúp SV làm quen với cách thức viết code để đọc dữ liệu ở dạng nhị phân từ file để xử lý/tính toán ngay bên trong máy mà không cần tốn chi phí mã hóa chúng, cách thức để ghi dữ liệu nhị phân bên trong chương trình ra file nhị phân cho hiệu quả cả về dung lượng chứa lẫn thời gian ghi tin (vì không cần tốn chi phí giải mã).

II. Nội dung :

- Viết code để đọc dữ liệu của 2 ma trận từ 2 file nhị phân, tính ma trận tổng, xuất kết quả ma trận tổng ra file nhị phân để đạt hiệu quả cao nhất.

III. Chuẩn đầu ra :

- Sinh viên nắm vững và lập trình thành thạo các đoạn code để đọc/ghi dữ liệu nhị phân từ file nhị phân vào các biến bên trong chương trình.

IV. Qui trình :

1. Vẫn giữ lại 2 file c:\A.txt và c:\B.txt chứa nội dung 2 ma trận A và B như sau :

Nội dung c:\A.txt như sau :
```
5, 7
1, 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
```

Và nội dung c:\B.txt như sau :
```
5, 7
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
```

2. Nhân bản thư mục chứa Project của bài thực hành 8.2 (thư mục TongMTTxt) thành thư mục mới và đặt tên cho thư mục này là TongMTBin.
3. Chạy VS .Net, mở lại Porject trong thư mục TongMTBin.
4. Hiệu chỉnh lại hàm WriteMT như sau :

```
//hàm ghi ma trận ra file binary
static void WriteMT(string path, double[,] A, int hang, int cot) {
    //1. tạo đối tượng quản lý file
    FileStream stream = new FileStream(path, FileMode.Create);
    //2. tạo đối tượng phục vụ ghi file
    BinaryWriter writer = new BinaryWriter(stream);
    //3. định nghĩa các biến dữ liệu theo yêu cầu chương trình
    int i, j;
    //4. ghi dữ liệu từ các biến ra file
    writer.Write(hang); //ghi số hàng
    writer.Write(cot); //ghi số cột
    //ghi ma trận
    for (i = 0; i < hang; i++) { //ghi từng hàng ma trận
        for (j = 0; j < cot; j++) {
```

```
                writer.Write(A[i, j]); //ghi phần tử i,j
            }
        }
        //5. đóng các đối tượng được dùng lại
        writer.Close(); stream.Close();
    }
```

5. Duyệt tìm hàm Main(), tìm và sửa lệnh gọi hàm WriteMT() thành :

```
    //xuất ma trận kết quả ra file c:\S.bin
    WriteMT("c:\\S.bin", S, hang, cot);
```

6. Chọn menu Debug.Start Debugging để dịch và chạy ứng dụng. Nếu không có lỗi thì chương trình sẽ chạy tốt và ghi nội dung ma trận tổng ra file S.bin. So sánh kích thước của file S.bin với file S.txt, lý giải tại sao kích thước của file S.bin chỉ có 288 byte.

7. Copy file c:\S.bin thành 2 file c:\A.bin và c:\B.bin để tạo 2 file miêu tả 2 ma trận A và B ở dạng nhị phân.

8. Hiệu chỉnh lại hàm ReadMT như sau :

```
//hàm đọc ma trận từ file binary
 static void ReadMT(string path, ref double[,] A, ref int hang, ref int cot) {
    //1. tạo đối tượng quản lý file
    FileStream stream = new FileStream(path, FileMode.Open);
    //2. tạo đối tượng phục vụ đọc file
    BinaryReader reader = new BinaryReader(stream);
    //3. định nghĩa các biến dữ liệu theo yêu cầu chương trình
    int i, j;
    //4. đọc dữ liệu từ file vào các biến
    //đọc số hàng
    hang = reader.ReadInt32();   //đọc số nguyên 32 bit
    //đọc số cột
    cot = reader.ReadInt32();      //đọc số nguyên 32 bit
    //phân phối vùng nhớ cho ma trận
    A = new double[hang, cot];
    //đọc từng phần tử ma trận
    for (i = 0; i < hang; i++)
       for (j = 0; j < cot; j++)
       {
          A[i, j] = reader.ReadDouble();   //đọc số thực
       }
    //5. đóng các đối tượng được dùng lại
    reader.Close(); stream.Close();
 }
```

9. Duyệt tìm hàm Main(), tìm và sửa lệnh 2 lệnh gọi hàm ReadMT() thành :

```
    //đọc ma trận A từ file c:\A.bin
    ReadMT("c:\\a.bin", ref A, ref hang, ref cot);
    //đọc ma trận B từ file c:\B.bin
    ReadMT("c:\\b.bin", ref B, ref h, ref c);
```

10. Duyệt tìm và xóa hàm ReadItem vì không cần thiết.

11. Chọn menu Debug.Start Debugging để dịch và chạy ứng dụng. Nếu không có lỗi thì chương trình sẽ chạy tốt, nó sẽ đọc lần lượt từng ma trận trong file nhị phân tương ứng vào chương trình, tình tổng ma trận và ghi nội dung ma trận tổng ra file S.bin.