LAB 5. CÁC KỸ THUẬT XỬ LÝ MẢNG MỘT CHIỀU

THỜI LƯỢNG: 8 TIẾT

A. Mục tiêu

- Giúp sinh viên hiểu rõ và thực hiện thuần thục các kỹ thuật xử lý trên mảng một chiều.
- Tiếp tục rèn luyện kỹ năng phát triển chương trình từng bước hoàn thiện chức năng.
- Sau khi hoàn thành bài thực hành , sinh viên :
  - Nắm vững các khái niệm và thao tác nhập, xuất trên mảng một chiều.
  - Nắm vững các kỹ thuật xử lý cơ bản trên mảng một chiều.
  - Biết cách định nghĩa và sử dụng kiểu dữ liệu mới bằng từ khóa **typedef**.
  - Hiểu rõ cơ chế gọi hàm và truyền tham số: truyền tham trị và truyền tham biến.
  - Thực hiện thuần thục cách tổ chức chương trình bằng thư viện hàm và menu.

B. Yêu cầu

- Nạp kết quả thực tập phần D (hướng dẫn thực hành) tại phòng Lab theo yêu cầu :
  - Tạo thư mục, đặt tên là **MSSV_Lab05_D_HD,** để lưu bài làm. Trong đó, MSSV là mã số của sinh viên.
  - Các bài 1,2 3, 4 : tạo các project theo hướng dẫn, lưu trử trong thu mục trên
  - Xóa thư mục Debug trong các project, nén thư mục và nạp cho giáo viên qua mail.
  - Thời gian nạp : Cuối buổi thực tập thứ 7
- Sinh viên tiếp tục nạp kết quả thực tập (phần E, bài tập bắt buộc) tại phòng Lab theo yêu cầu :
  - Tạo project đặt tên là **MSSV_Lab05_E_BB,** có các chức năng được lấy trong các chức năng của 7 bài tập phần E, mỗi bài chọn tùy ý 2 chức năng.
  - Xóa thư mục Debug trong các project, nén thư mục và nạp cho giáo viên qua mail.
  - Thời gian nạp : Cuối buổi thực tập thứ 8.

- Giải bài tập lab 5 :
  - Tiết 1,2 buổi thực tập thứ 9.

C. Ôn tập lý thuyết

1. Cú pháp khai báo (định nghĩa) mảng một chiều

Cú pháp: **KDL Tên_biến_mảng [Kích_thước];**

Trong đó:

- **KDL**: là kiểu dữ liệu của các phần tử chứa trong mảng.
- **Tên_biến_mảng**: là tên của mảng, do người lập trình tự đặt và phải tuân theo quy tắc đặt tên.
- **Kích_thước**: là một số nguyên dương, cho biết số phần tử tối đa có thể chứa trong mảng.

2. Cú pháp định nghĩa kiểu dữ liệu mảng 1 chiều

Cú pháp: **typedef KDL Tên_kiểu_mảng [Kích_thước];**

Trong đó:

- **KDL**: là kiểu dữ liệu của các phần tử chứa trong mảng.
- **Tên_kiểu_mảng**: là tên của kiểu dữ liệu (mới) mảng một chiều.

3. Các thao tác nhập - xuất mảng một chiều

a. Trường hợp không sử dụng kiểu mảng một chiều

```
// Định nghĩa hàm nhập giá trị các phần tử của mảng
// bằng cách nhập lần lượt từ bàn phím.
// Input : a = mảng một chiều chứa tối đa MAX phần tử.
//         n = số phần tử thực sự được lưu trong mảng.
// Output: Không có.
void NhapMang(int a[MAX], int n)
{
    // Duyệt qua từng phần tử từ vị trí 0 tới n-1
    for (int i=0; i<n; i++)
    {
        // Xuất thông báo yêu cầu người dùng nhập
        cout << "a[" << i << "] = ";

        // Chờ người dùng nhập phần tử thứ i
        cin >> a[i];
    }
}
// Định nghĩa hàm nhập giá trị cho các phần tử của
// mảng bằng cách sinh các số ngẫu nhiên
// Input : a = mảng một chiều chứa tối đa MAX phần tử.
//         n = số phần tử thực sự được lưu trong mảng.
// Output: Không có.
void NhapTuDong(int a[MAX], int n)
{
    // Gieo số ngẫu nhiên đầu tiên
    srand(time_t(NULL));

    // Duyệt qua từng phần tử từ vị trí 0 tới n-1
    for (int i=0; i<n; i++)
    {
        // Sinh một số ngẫu nhiên trong phạm vi
        // [0..MAX) rồi gán cho phần tử thứ i
        a[i] = rand() % MAX;
    }
}
// Định nghĩa hàm xuất các phần tử của mảng ra màn hình
// Input : a = mảng một chiều chứa tối đa MAX phần tử.
//         n = số phần tử thực sự được lưu trong mảng.
// Output: Không có. Chỉ xuất ra màn hình.
void XuatMang(int a[MAX], int n)
{
    cout << endl << "Cac phan tu cua mang : " << endl;

    for (int i=0; i<n; i++)
        cout << a[i] << TAB;

    cout << endl << endl;
}
```

Lưu ý quan trọng:

- Đối với mảng 1 chiều, có 2 giá trị thường đi kèm: **MAX** và **n**.

- **MAX**: là kích thước khai báo, là số phần tử tối đa mảng có thể chứa. Giá trị này phải xác định trước và thường được định nghĩa là một hằng số.
- **n**: là số phần tử thực sự chứa trong mảng (n < MAX) và giá trị này thay đổi trong mỗi lần chạy chương trình.

- Truyền tham số:

Tham số hình thức (tên khai báo lúc định nghĩa hàm)	**Đối số (giá trị truyền vào lúc gọi hàm)**
Tên mảng một chiều, viết 1 trong 2 dạng sau: **int a[MAX]** hoặc **DaySo a** trong đó: DaySo là kiểu dữ liệu mảng 1 chiều.	Tên của mảng một chiều có cùng kích thước, cùng kiểu với tham số hình thức.

b. Trường hợp sử dụng kiểu dữ liệu mảng một chiều

Trước hết, cần định nghĩa kiểu dữ liệu mảng một chiều. Đặt tên kiểu dữ liệu mới là DaySo.

```
9   // Định nghĩa kiểu dữ liệu mảng 1 chiều
10  typedef     int     DaySo[MAX];
```

Sau đó, thay thế các tham số **int a[MAX]** trong ba hàm nhập-xuất ở trên bởi **DaySo a**. Phần nội dung bên trong hàm không thay đổi.

```
16  void NhapMang(DaySo a, int n);
17  void NhapTuDong(DaySo a, int n);
18  void XuatMang(DaySo a, int n);
```

Với cách này, nếu cần thay đổi kiểu dữ liệu của các phần tử trong mảng hoặc kích thước mảng, ta chỉ cần sửa đổi mã lệnh ở dòng định nghĩa kiểu dữ liệu mảng (lệnh typedef).

4. Các kỹ thuật xử lý mảng một chiều

a. Kỹ thuật thử - sai

Áp dụng khi cần xác định Kq (kết quả) và biết Kq ∈ { a_0, a_1, …, a_{n-1} }. Cách thực hiện như sau:

- Giả sử Kq = a_0
- Duyệt các phần tử còn lại để thử và xác định chính xác giá trị của Kq.

b. Kỹ thuật duyệt

- Duyệt toàn cục: duyệt hết tất cả các phần tử của mảng.
- Duyệt cục bộ: chỉ xét một phần của mảng.

c. Kỹ thuật kiểm tra tính đúng - sai

- Bài toán AND:
  - Dạng:
    - Đúng: nếu với mọi i, a_i đều thỏa mãn điều kiện.
    - Sai: nếu tồn tại i sao cho a_i không thỏa mãn điều kiện.
  - Cách thực hiện:
    - Gán Kq = 1; // Giả sử kết quả là đúng
    - Duyệt để tìm điều kiện sai. Nếu có gán Kq = 0 và dừng.
- Bài toán OR
  - Dạng:
    - Đúng: nếu tồn tại i sao cho a_i thỏa mãn điều kiện.
    - Sai: nếu với mọi i, a_i đều không thỏa mãn điều kiện.
  - Cách thực hiện:
    - Gán Kq = 0; // Giả sử kết quả là sai

- Duyệt để tìm điều kiện đúng. Nếu có gán Kq = 1 và dừng.

D. Hướng dẫn thực hành

Phần này xây dựng chương trình thực hiện các thao tác trên cấu trúc dữ liệu mảng 1 chiều, minh họa các kỹ thuật xử lý mảng 1 chiều trong các bài toán tìm kiếm, tìm Max, Min, đếm, tính tổng, tích, sắp xếp, . . . Chương trình tổ chức theo thư viện và có/không có hệ thống menu (theo mục 1 hay 2 phần C- hướng dẫn - lab 4). Ngoài ra, khi soạn thảo chương trình tiếp tục rèn luyện cách viết có cấu trúc về mặt hình thức.

Bài 1: **Chương trình minh họa bài toán AND, OR, kỹ thuật thử và sai.**

Viết chương trình thực hiện các thao tác trên dãy a gồm n số nguyên. Chương trình cho phép người dùng chọn các chức năng từ menu sau:

1. Kiểm tra phần tử x có trong dãy a không? Nếu có, trả về 1. Nếu không, trả về 0.
2. Tìm vị trí xuất hiện đầu tiên của phần tử ***x*** trong mảng ***a***. Nếu a không chứa ***x***, trả về -1.
3. Kiểm tra mảng a có thứ tự tăng?
4. Tìm phần tử có giá trị lớn nhất
5. Tìm vị trí đầu tiên của phần tử lớn nhất
6. Kiểm tra phát biểu : nếu a chứa x thì a cũng chứa –x.

Bước 1. Tạo dự án Win32 Console Application mới. Đặt tên là **Lab05_D_Bai1**

Bước 2. Tạo cấu trúc cho chương trình như đã hướng dẫn trong **mục 2 lab 4** (từ 1- 8 để có cấu trúc nội dung tối thiểu chạy được chương trình).

Bước 3:

- **Trong tập tin *thuvien.h* :**

Ta bổ sung định nghia hằng, kiểu dũ liệu mới :

Vì chương trình thực hiện các thao tác trên mảng 1 chiều, nên ta cần định nghĩa một hằng là giá trị kích thước khai báo của mảng. Ngoài ra, ta có thể định nghĩa một kiểu dữ liệu mảng 1 chiều (lưu ý rằng có thể không cần thực hiện định nghĩa này vì ta có thể làm trực tiếp trên biến mảng)

```
//Dinh nghia hang
#define MAX 100 //kich thuoc khai bao mang 1chieu
#define   TAB  '\t'

//Dinh nghia kieu du lieu moi:
typedef  int  DaySo[MAX];

//Khai bao nguyen mau cac ham xu ly, nhap xuat
        //bổ sung sau

//Dinh nghia cac ham xu ly, nhap xuat
        //bổ sung sau
```

- **Trong tập tin *menu.h* :**

ta viết lại như sau (cấu trúc giống như bước 9 mục 2 lab 4, chỉ thay đổi nội dung theo yêu cầu bài toán) :

```
// Khai báo nguyên mẫu các hàm xử lý menu
        //bổ sung sau
 // Định nghĩa các hàm xử lý menu
```

3.1 Định nghĩa hàm xuất danh sách chức năng ra màn hình

```
// Định nghĩa hàm xuất danh sách chức năng ra màn hình
//Ngoai cac chuc nang cua bai toan, ta them chuc nang xem du lieu day so
// Input : Không có
// Output: Không có
void XuatMenu()
{
	cout << endl << "============== CHON CHUC NANG =============";
	cout << endl << "0. Thoat khoi chuong trinh";
	cout << endl << "1. Kiem tra x nam trong mang a";
	cout << endl << "2. Tim vi tri dau tien x xuat hien trong a";
	cout << endl << "3. Kiem tra mang a la day tang";
	cout << endl << "4. Tim phan tu lon nhat";
	cout << endl << "5. Tim vi tri cuoi cung gia tri lon nhat xuat hien";
	cout << endl << "6. Neu a chua x thi cung chua -x";
	cout << endl << "7. Xem du lieu day so";
	cout << endl << "========================================";
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

3.2 Định nghĩa hàm chọn một menu trong danh sách

```
// Input : soMenu = Số lượng menu có thể chọn.
// Output: Số thứ tự menu do người dùng nhập vào.
int ChonMenu(int soMenu)
{
	int stt;
	for (;;)
	{
		system("CLS");
		XuatMenu();
		cout<<"\nNhap 1 so khong khoang [0,.," << soMenu << "] de chon chuc nang, stt = ";
		cin >> stt;
		if (0 <= stt && stt <= soMenu)
			break;
	}
	return stt;

}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

3.3 Định nghĩa hàm xử lý menu :

Các thao tác đều thực hiện trên cùng một đầu vào là mảng 1 chiều kiểu DaySo, nên ta bổ sung thêm biến mảng a kiểu DaySo với kích thước mảng thực dùng trong mỗi lần thực hiện chương trình là số nguyên dương n làm đối của hàm **XuLyMenu**, ngoài tham số đã có là tham số menu.

```
// Input : menu = Số thứ tự menu do người chọn,
//                Dãy số a
//                số nguyên dương n
// Output: Không có.
void XuLyMenu(int menu, DaySo a, int n)
{
	// Khai báo biến
```

```
	switch (menu)
	{

		case 0:
			system("CLS");
			cout << endl<<"\n0. Thoat khoi chuong trinh\n";
			break;
		case 1:
			system("CLS");
			cout << endl << "1. Kiem tra x nam trong mang a";
			//Bo sung sau
			break;
		case 2:
			system("CLS");
			cout << endl << "2. Tim vi tri dau tien x xuat hien trong a";
			//Bo sung sau
			break;
		case 3:
			system("CLS");
			cout << endl << "3. Kiem tra mang a la day tang";
			//Bo sung sau
			break;
		case 4:
			system("CLS");
			cout << endl << "4. Tim phan tu lon nhat";
			//Bo sung sau
			break;

		case 5:
			system("CLS");
			cout << endl << "5. Tim vi tri phan tu lon nhat";
			//Bo sung sau
			break;
		case 6:
			system("CLS");
			cout << endl << "6. Neu a chua x thi cung chua -x";
			//Bo sung sau
			break;
		case 7:
			system("CLS");
			cout << endl << "7. Xem du lieu day so";
			//Bo sung sau
			break;
	}
	_getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

3.4 Bổ sung khai báo nguyên mẫu các hàm tổ chức menu trong phần khai báo nguyên mẫu hàm

```
void XuatMenu();
int ChonMenu(int soMenu);
void XuLyMenu(int menu, DaySo a, int n);
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có

- Trong tập tin ***program.cpp*** :
  Hàm ***ChayChuongTrinh*** ta cập nhật lại như sau :

```
void ChayChuongTrinh()
{
	int soMenu = 7, //lưu số các chức năng
		menu, // lưu số thứ tự chức năng người dùng chọn
		n=0; //kích thước mảng và giá trị khởi tạo
	DaySo a;
	do
	{
		menu = ChonMenu(soMenu);
		XuLyMenu(menu,a,n);
	} while (menu > 0);

}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.
Kiểm tra chức năng thoát khỏi chương trình (chọn 0)

Bước 4 :

Trong bước 4 này, ta làm công việc sau :

- Trong tập tin ***thuvien.h*** , soạn thảo các hàm nhập, xuất dãy số
- Trong tập tin ***menu.h*** bổ sung xử lý chức năng xem dữ liệu trong hàm ***XuLyMenu.***
- Trong tập tin ***program.cpp*** cập nhật lại nội dung hàm ***ChayChuongTrinh*** : Nhập dữ liệu cho dãy số, điều khiển tùy chọn thực hiện menu chương trình (kiểm tra chức năng 7 - chọn 7 để xem dữ liệu, chọn từ 1 đến 6 thì chưa làm gì cả).

- Trong tập tin *thuvien.h* :

Bổ sung các hàm nhập xuất :

4.1 Hàm nhập dữ liệu mảng 1 chiều từ bàn phím

```
// Input : a = mảng một chiều chứa tối đa MAX phần tử.
//	n = số phần tử thực sự được lưu trong mảng.
// Output: Không có.
void NhapMang(DaySo a, int n)
{
	int i;
	// Duyệt qua từng phần tử từ vị trí 0 tới n-1
	for (i=0; i<n; i++)
	{
		// Xuất thông báo yêu cầu người dùng nhập
		cout << "a[" << i << "] = ";
		// Chờ người dùng nhập phần tử thứ i
		cin >> a[i];
	}
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

4.2 Định nghĩa hàm xuất các phần tử của mảng ra màn hình

```
//Input : a = mảng một chiều chứa tối đa MAX phần tử.
//	n = số phần tử thực sự được lưu trong mảng.
```

```
// Output: Không có. Chỉ xuất ra màn hình.
void XuatMang(DaySo a, int n)
{
	int i;
	for (i=0; i<n; i++)
		cout << a[i] << TAB; //hai gia tri cach nhau 1 tab
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

4.3 Bổ sung nguyên mẫu các hàm :

```
void NhapMang(DaySo a, int n);
void XuatMang(DaySo a, int n);
```

- Trong tập tin *menu.h* :

Bổ sung xử lý chức năng xem dữ liệu của dãy số trong case 7 (Các case từ 0 đến 6 giữ nguyên, bổ sung xử lý trong case 7)

```
void XuLyMenu(int menu, DaySo a, int n)
{
	// Khai báo biến
	switch (menu)
	{

		case 0:
			system("CLS");
			cout <<endl<< "\n0. Thoat khoi chuong trinh\n";
			break;
		//. . .
		case 7:
			system("CLS");
			cout << endl << "7. Xem du lieu day so";
			cout << "\nDay so hien hanh:\n";
			XuatMang(a, n);
			break;
	}
	_getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

- Trong tập tin program.cpp :

4.4 **Cập nhật lại nội dung hàm *ChayChuongTrinh()* :**

Nhập dữ liệu cho mảng để chương trình xử lý theo menu

```
void ChayChuongTrinh()
{
	// Khai bao bien
	int menu,
		soMenu = 7,
		n = 0;
	DaySo a;
	cout << endl << "Nhap mot so nguyen duong : ";
```

```
	cin >> n;
	//Nhập dữ liệu cho mang a
	NhapMang(a, n);
	// Lặp lại việc chọn và xử lý menu cho tới khi
	//người dùng chọn chức năng 0. Thoát khỏi CT.
	do
	{
		menu = ChonMenu(soMenu);
		XuLyMenu(menu,a,n);
	} while (menu > 0); //menu =0 thi dung chuong trinh
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.
Kiểm tra thực hiện việc chọn chức năng 7 để xem dữ liệu.

Trong các bước tiếp theo, ta cập nhật chương trình bằng cách bổ sung và hoàn thiện từng chức năng vào chương trình :

- Lần lượt soạn thảo từng hàm chức năng trong tập tin ***thuvien.h***,
- Lần lượt bổ sung xử lý chức năng trong hàm ***XuLyMenu*** của ***menu.h***,

Bước 5: Bổ sung chức năng 1 (kiểm tra a có chứa x) chương trình..

- Trong ***thuvien.h***,:

5.1 Định nghĩa hàm kiểm tra mảng a có chứa phần tử x?

```
// Input :  a  - mảng một chiều chứa tối đa MAX phần tử.
//          n  - số phần tử thực sự được lưu trong mảng.
//          x  - phần tử cần kiểm tra
// Output:
//          1 : nếu mảng a chứa phần tử x
//          0 : nếu mảng a không chứa phần tử x

int ChuaX(DaySo a, int n, int x)
{
	int i, kq;
	kq = 0; // Ban đầu, giả sử mảng a không chứa x
	// Duyệt qua các phần tử để kiểm tra
	for (i=0; i<n; i++)
		if (a[i] == x)		// Nếu tìm thấy phần tử x
		{
			kq = 1;		// thì cập nhật kết quả và
			break;		// dừng, không cần tìm nữa
		}
	return kq;
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

5.2 - Khai báo nguyên mẫu hàm :

```
int ChuaX(DaySo a, int n, int x);
```

- Trong ***menu.h*** :

Nội dung hàm XuLyMenu bổ sung khai báo biến x kiểu int (để lưu trử giá trị cần tìm được nhập từ bàn phím), biến kq kiểu int để lưu trử két quả tìm kiếm, bổ sung việc thực hiện chức năng 1 trong case 1, các case khác giữ nguyên.

```
void XuLyMenu(int menu, DaySo a, int n)
{
	// Khai báo biến
	int x, kq;
	switch (menu)
	{


		//. . .
		case 1:
			cout << endl << "1. Kiem tra x nam trong mang a";
			cout << endl << "Nhap gia tri x : ";
			cin >> x;
			kq = ChuaX(a, n, x);
			// Xuất thông báo
			system("CLS");
			cout << "\nMang hien hanh:\n";
			XuatMang(a, n);
			if (kq)
				cout << endl << "Mang co chua " << x;
			else
				cout << endl << "Mang khong chua " << x;
			break;
		//. . .
	}
	_getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.
Kiểm tra thực hiện chức năng 1 (kiểm tra mảng có chữa x hay không ?).

Bước 6: Bổ sung chức năng 2 (tìm vị trí đầu tiên x xuất hiện trong a) vào chương trình.
- Trong ***thuvien.h*** :

6.1 Định nghĩa hàm tìm vị trí đầu tiên x xuất hiện trong a.?

```
// Input : a,  n,x,
//Output:
	-1 : nếu a không chứa phần tử x
	 i : a[i] đầu tiên trùng x

int Tim_VTDT_X(DaySo a, int n, int x)
{
	int i,
	kq = -1; //Ban đầu, giả sử mảng a không chứa x

	// Duyệt qua các phần tử để kiểm tra
	for (i=0; i<n; i++)
		if (a[i] == x)		// Nếu tìm thấy phần tử x
		{
			kq = i;		// thì cập nhật kết quả là vị trí đầu tiên là I và
			break;		// dừng, không cần tìm nữa
		}
	return kq;
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có

6.2 Khai báo nguyên mẫu hàm :

```
int Tim_VTDT_X(DaySo a, int n, int x);
```

- Trong ***menu.h*** :

Bổ sung việc thực hiện chức năng 2 trong case 2, các case khác giữ nguyên.

```
void XuLyMenu(int menu, DaySo a, int n)
{
	// Khai báo biến
	int x, kq;
	switch (menu)
	{
		//. . .
		case 2:
			cout << endl << "2. Tim vi tri dau tien x xuat hien trong a";
			cout << endl << "Nhap gia tri x : ";
			cin >> x;
			kq = Tim_VTDT_X(a, n, x);
			// Xuất thông báo
			system("CLS");
			cout << "\nMang hien hanh:\n";
			XuatMang(a, n);
			if (kq == -1)
				cout << endl << "Mang khong chua " << x;
			else
				cout << "\nVi tri dau tien " << x<<" xuat hien trong a la : "<<kq;
			break;
		//. . .
	}
	_getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có..

Kiểm tra kết quả thực hiện chức năng 2.

Bước 7: Bổ sung chức năng 3(kiểm tra mảng a có tăng) vào chương trình.

- Trong ***thuvien.h***,

7.1 Định nghĩa hàm kiểm tra mang a có tăng ?

```
// Input : a , n
// Output:
//		1 : nếu mảng a có thứ tự tăng
//		0 : nếu mảng a không có thứ tự tăng

int KiemTraMangTang(DaySo a, int n)
{
	int i,
	kq = 1; // Ban đầu, giả sử mảng a có thứ tự tăng

	// Duyệt qua các phần tử để kiểm tra
	for (i=0; i<n-1; i++)
		if (a[i] > a[i+1])// Nếu có cặp phần tử mà số
		{			// đứng trước > số đứng sau
```

```
                        kq = 0;  // thì cập nhật kết quả và
                        break;   // dừng, không cần ktra nữa
                }
        return kq;
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

7.2 Khai báo nguyên mẫu hàm :

```
 int KiemTraMangTang(DaySo a, int n);
```

- Trong ***menu.h*** :

Bổ sung xử lý chức năng 3 trong case 3, các case khác giữ nguyên.

```
void XuLyMenu(int menu, DaySo a, int n)
{
        // Khai báo biến
        int x, kq;
        switch (menu)
        {


                //. . .
                case 3:
                        cout << endl << "3. Kiem tra mang a tang";
                        kq =  KiemTraMangTang(a,n);
                        // Xuất thông báo
                        system("CLS");
                        cout << "\nMang hien hanh:\n";
                        XuatMang(a, n);
                        if (kq)
                                cout << endl << "a La Mang tang.";
                        else
                                cout << endl << "a không phai Mang tang.";
                        break;
                //. . .
        }
        _getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có..

Kiểm tra kết quả thực hiện chức năng 3.

Bước 8: Bổ sung chức năng 4(tìm giá trị lơn nhất) vào chương trình.

- Trong ***thuvien.h***,

8.1 Đinh nghĩa hàm tính Max

```
// Input : a, n
// Output: Giá trị lớn nhất của a

int TinhMax(DaySo a, int n)
{
        int i,
           max; //luu tru gia tri lon nhat cua mang
        max = a[0]; // Ban đầu, giả sử phần tử đầu tiên là lớn nhất
        // Duyệt qua các phần tử để kiểm tra giả thuyết
```

```
	for (i=1; i<n; i++)
		if (a[i] > max)		// Nếu có phần tử lớn hơn
			max = a[i];		// pt giả thuyết, cập nhật
	return max;
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

8.2 Khai báo nguyên mẫu hàm :

```
int TinhMax(DaySo a, int n);
```

- Trong ***menu.h*** :

Bổ sung xử lý chức năng 4 trong case 4, các case khác giữ nguyên.

```
void XuLyMenu(int menu, DaySo a, int n)
{
	// Khai báo biến
	int x, kq;
	switch (menu)
	{
		//. . .
		case 4:
			system("CLS");
			cout << endl << "4. Tim phan tu lon nhat";
			kq = TinhMax(a, n);
			// Xuất thông báo
			system("CLS");
			cout << "\nMang hien hanh:\n";
			XuatMang(a, n);
			cout << "\nMax[0,..," << n << "] = " << kq;
			break;
		//. . .
	}
	_getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có..

Kiểm tra kết quả thực hiện chức năng 4.

Bước 9: Bổ sung chức năng 5 (tìm vị trí xuất hiện cuối cùng của giá trị max) vào chương trình.

- Trong ***thuvien.h***,

9.1 Đinh nghĩa hàm tìm vị trí xuất hiện cuối cùng của giá trị max

```
//Ham tim vi tri cuoi cung max xuat hien, khong dung ham TinhMax
// Input : a, n
// Output: Vị trí cuối cùng tìm thấy giá trị lớn nhất

int TimViTriMax_CuoiCung(DaySo a, int n)
{
	// Ban đầu, giả sử phần tử đầu tiên là lớn nhất
	int vt = 0,
		max = a[vt];
	int i;
	// Duyệt qua các phần tử để kiểm tra giả thuyết
	for (i = 1; i<n; i++)
	if (a[i] >= max)		// Nếu có phần tử khong nho hơn
	{				// giá trị max giả định
```

```
                vt = i;             // thì cập nhật lại vị trí
                max = a[vt];        // và phần tử lớn nhất
        }
        return vt;
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có

9.2 Khai báo nguyên mẫu hàm :

```
int TimViTriMax_CuoiCung(DaySo a, int n);
```

- Trong ***menu.h*** :

Bổ sung xử lý chức năng 5 trong case 5, các case khác giữ nguyên.

```
void XuLyMenu(int menu, DaySo a, int n)
{
        // Khai báo biến
        int x, kq;
        switch (menu)
        {
                //. . .
                case 5:
                        system("CLS");
                        cout << endl << "5. Tim vi tri cuoi cung gia tri lon nhat";
                        kq = TimViTriMax_CuoiCung(a, n);

                        // Xuất thông báo
                        system("CLS");
                        cout << "\nMang hien hanh:\n";
                        XuatMang(a, n);
                        cout << endl << "Vi tri xuat hien cuoi cung cua gia tri lon nhat la "
                                << kq;
                        break;
                //. . .
        }
        _getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có..

Kiểm tra kết quả thực hiện chức năng 5.

Bước 10: Bổ sung chức năng 6 (nếu a chứa x thì cũng chứa –x) vào chương trình..

- Trong ***thuvien.h***,

10.1 Đinh nghĩa hàm kiểm tra phát biêu Nếu a chứa x thì cũng chứa -x

```
//Chi cần xét các phần tử trong mảng .
// Input : a , n
// Output:      1; nếu đúng;  // ∀i :-a[i] ∈ a
                0,  nếu sai. //  ∃i: -a[i] ∉ a
//Chi quan tam toi cac phan tu trong a
int ChuaXChuaTruX(DaySo a, int n)
{
        int  i,//duyet cac phan cua mang
                j,//duyet cac phan tu sau a[i] de xem co bang -a[i]
                kq, // Luu ket qua kiem tra phat bieu
```

```
                x, //luu gia tri a[i]
                kqTam; //luu ket qua kiem tra a co chua -x
        kq = 1; //dau tien xem ket qua phat bieu la dung : moi i, a deu chua –a[i]
        // Duyệt qua các phần tử để kiểm tra
        for (i = 0; i < n ; i++)
        {
                x = a[i]; // luu a[i] vao x
                kqTam = 0; //dau tien cho rang mang khong chua -x
                for (j = 1; j < n; j++)
                        if (a[j] == -x)
                        {
                                kqTam = 1; //cap nhat ket qua a co chua -x
                                break;
                        }
                if (kqTam == 0)  //neu a khong cua -x
                {
                        kq = 0; //phat bieu sai
                        break;
                }
        }
        return kq;
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

10.2 Khai báo nguyên mẫu hàm :

```
int ChuaXChuaTruX(DaySo a, int n);
```

- Trong ***menu.h*** :

Bổ sung việc chức năng 6 trong case 6, các case khác giữ nguyên.

```
void XuLyMenu(int menu, DaySo a, int n)
{
        // Khai báo biến
        int x, kq;
        switch (menu)
        {
                //. . .
                case 6:
                        system("CLS");
                        cout << endl << "6. Neu a chua x thi cung chua -x";
                        kq = ChuaXChuaTruX(a, n);

                        // Xuất thông báo
                        system("CLS");
                        cout << "\nMang hien hanh:\n";
                        XuatMang(a, n);

                        if (kq)
                                cout << endl << "phat bieu dung ";
                        else
                                cout << endl << "phat bieu sai ";
                        break;
                        //. . .
```

```
	}
	_getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có..
Kiểm tra kết quả thực hiện chức năng 6.
Kiểm tra lại tất cả các chức năng – Kết thúc chương trình.

//

Ghi chú 1:

Trong chương trình trên, dữ liệu được nhập trong hàm ChayChuongTrinh, truyền đến hàm XuLyMenu qua đối a và n của hàm. Các chức năng thực hiện qua bộ dữ liệu này.
Khi đang thực hiện chương trình, các chức năng muốn thực hiện trên một dữ liệu khác thì không được. Ta chỉ có thể chạy lại chuong trình và nhập liệu lại.
Để linh hoạt hơn, ta bổ sung thêm một chức năng mới, chức năng thứ 8, chọn lại bộ dữ liệu mới.

Chương trình bổ sung như sau :
- Trong ***thuvien.h*** ta đã có hàm nhập dữ liệu cho mảng .
- Trong ***menu.h*** :

+ Hàm XuatMenu, bổ sung thêm tên chức năng thú 8 :

```
void XuatMenu()
{
	cout << endl << "=========================================";

		//tên các chức năng từ 0 đến 7 như cũ
	cout << endl << "8. Chon lai bo du lieu moi cho day so";
	cout << endl << "=========================================";
}
```

+ Hàm XuLyMenu bổ sung xử lý chức năng 8 trong case 8:

```
void XuLyMenu(int menu, DaySo a, int n)
{
	// Khai báo biến
	int x,//gia tri tim kiem
		kq;//ket qua tim kiem
	switch (menu)
	{

		//. . .
		case 8:
			system("CLS");
			cout << endl << "8. Chon bo du lieu khac";
			cout << "\nNhap lai kich thuoc n : ";
			cin >> n;
			//goi ham nhap du lieu
			NhapMang(a, n);
			system("CLS");
			cout << "\nDay so moi nhap:\n";
			XuatMang(a, n);
			break;

	}
	_getch();
}
```

- Trong tập tin ***program.cpp***, Hàm ChayChuongTrinh sửa lại gái trị soMenu, soMenu = 8.

Nhấn Ctrl + F5 để chạy chương trình thực hiện chức 8 để cập lại bộ dữ liệu mới nhập cho chương trình, Sử dụng bộ dữ liệu mới này cho các chức năng khác ! Không có được kết quả mong muốn.

Đó là vì khi thực hiện xong case 8, bộ dữ liệu mới nhập không lưu lại được nên không thây thay thế được bộ dữ liệu cũ (nói chung, với cách viết này, nếu dữ liệu của a có thay đổi trong các case của hàm xử XuLyMenu thì thì không lưu giữ lại được để dung cho các case khác).
Để khắc phục điều này, trong đối n của hàm XuLyMenu ta sẽ dùng tham chiếu, tức là khi đó n xem như đối ra, cách truyền tham số là truyền bằng biến.

Dòng tiêu đề của hàm XuLyMenu viết lại :

```
void XuLyMenu(int menu, DaySo a, int &n)
```

Nhấn Ctrl + F5 để chạy chương trình, kiểm tra kết quả.
Từ đây vê sau, nếu các chức năng của chương trình đều thực hiện trên mảng a kích thước n, và làm thay đổi dữ liệu của a, thì đối n trong hàm XuLyMenu sẽ được viết dưới dạng tham chiếu.

Ghi chú 2:
Ta có thể nhập dữ liệu trong hàm XuLyMenu (như một chức năng chương trình, đưa vào xử lý trong câu lệnh switch).
Ghi chú 3:
Để tiết kiệm thời gian cho việc nhập mảng từ bàn phím, ta viết hàm nhập liệu tự động trong ***thuvien.h*** như sau :

```
void NhapTuDong(DaySo a, int n)
{
	int i;
	// Gieo số ngẫu nhiên đầu tiên
	srand(time(NULL));
	for (i = 0; i<n; i++)
	{
		// Sinh một số ngẫu nhiên trong phạm vi
		// [0..MAX) rồi gán cho phần tử thứ i
		a[i] = rand() % MAX;
	}
}
```

Khi đó :
- Thay thế hàm NhapMang bằng hàm NhapTuDong trong chương trình.
- Trong tập tin ***program.cpp*** ta khai báo bổ sung các thư viện **<time.h>, <stdlib.h>**

Ghi chú 4:
Trong chương trình nếu không định nghĩa kiêu Dayso, ta có thể định nghĩa trực tiếp biến mảng :

```
int a[MAX];
```

//

Bài 2: Minh họa các bài tính toán trên dãy số

Viết chương trình thực hiện các thao tác trên dãy a gồm n số nguyên. Chương trình yêu cầu nhập dữ lieu cho a và cho phép người dùng chọn các chức năng trong menu:

- Đếm số lần xuất hiện của giá trị x trong a.
- Đếm và xuất các số nguyên tố trong a.
- Tính tổng giá trị cac phàn tư trong a

- Tính tổng các giá trị chỉ xuất hiện một lần trong dãy.
- Tính tổng các giá tri phân biệt trong dãy.

Trong bài này, ta không nhập dữ liệu từ hàm ChayChuongTrinh để truyền đến hàm XuLyMenu qua các đối a và n, mà ta sẽ nhập trong hàm XuLyMenu, xem như là một chức năng của chương trình. Vấn đề là lần đầu tiên thực hiện chức năng của chương trình, ta cần chọn trước chức năng nhập dữ liệu cho mảng a, sau đó mới chọn thực hiên các chức năng khác, khi đó các chức năng sẽ thao tác trên bộ dữ liệu đã nhập của a. Nếu chọn lại nhập dữ liệu, tức là ta muốn thức hiện các thao tác trên bộ dữ liệu mới.

Bước 1. Tạo dự án Win32 Console Application mới. Đặt tên là **Lab05_D_Bai2**
Bước 2. Tạo cấu trúc cho chương trình như đã hướng dẫn trong **mục 2 lab 4** (từ 1- 8 để có cấu trúc nội dung tối thiểu chạy được chương trình).
Bước 3:

- Trong tập tin ***thuvien.h***, ta bổ sung định nghia hằng, kiểu dũ liệu mới :

```
//Dinh nghia hang, kieu du lieu moi
//Dinh nghia hang
#define MAX 100 //kich thuoc khai bao mang 1chieu
#define   TAB  '\t'

//Dinh nghia kieu du lieu moi:
typedef int DaySo[MAX];

//khai bao nguyen mau cac ham xu ly, nhap xuat


//Dinh ngia cac ham xu ly, nhap xuat
```

- Trong tập tin ***menu.h*** ta viết lại như sau (cấu trúc giống như bước 9 mục 2 lab 4, chỉ thay đổi nội dung theo yêu cầu bài toán) :

```
// Khai báo nguyên mẫu các hàm xử lý menu
//bổ sung sau

// Định nghĩa các hàm xử lý menu

3.1 Định nghĩa hàm xuất danh sách chức năng ra màn hình
void XuatMenu()
{
	cout << endl << "============== CHON CHUC NANG =============";
	cout << endl << "0. Thoat khoi chuong trinh";
	cout << endl << "1. Nhap tu dong mang a";
	cout << endl << "2. Xem du lieu mang a";
	cout << endl << "3. Dem so lan xuat hien cua x trong a";
	cout << endl << "4. Dem va xuat cac so nguyen to trong a";
	cout << endl << "5. Tinh tong cac gia tri  trong mang";
	cout << endl << "6. Tính tong cac gia tri chi xuat hien mot lan trong mang";
	cout << endl << "7. Tinh tong cac gia tri phan biet trong mang";
	cout << endl << "=========================================";
}
	Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

3.2 Định nghĩa hàm chọn một menu trong danh sách
```

```
// Input : soMenu = Số lượng menu có thể chọn.
// Output: Số thứ tự menu do người dùng nhập vào.
int ChonMenu(int soMenu)
{
	int stt;
	for (;;)
	{
		system("CLS");
		XuatMenu();
		cout<<"\nNhap 1 so khong khoang [0,.," << soMenu << "] de chon chuc nang, stt = ";
		cin >> stt;
		if (0 <= stt && stt <= soMenu)
			break;
	}
	return stt;

}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

3.3 Định nghĩa hàm xử lý menu :

Chú ý rằng các thao tác đều thực hiện trên cùng một đầu vào là mảng kiểu DaySo, nên ta bổ sung thêm biến mảng a kiểu DaySo với kích thước mảng là số nguyên dương n làm đối của hàm **XuLyMenu** (ngoài tham số menu). Để lưu giữ sự thay đổi dữ liệu trong a, n sẽ được viết dạng tham chiếu.

```
// Input : menu = Số thứ tự menu do người chọn,
                  Dãy số a
//                 số nguyên dương n
// Output: Không có.
void XuLyMenu(int menu, DaySo a, int &n)
{
	// Khai báo biến

	switch (menu)
	{

		case 0:
			system("CLS");
			cout << endl<<"\n0. Thoat khoi chuong trinh";
			break;
		case 1:
			system("CLS");
			cout << endl << "1. Nhap tu dong mang a";
			//Bo sung sau
			break;
		case 2:
			system("CLS");
			cout << endl << "2. Xem du lieu mang a";
			//Bo sung sau
			break;
		case 3:
			system("CLS");
			cout << endl << "3. Dem so lan xuat hien cua x trong a";
			//Bo sung sau
			break;
```

```
		case 4:
			system("CLS");
			cout << endl << "4. Dem va xuat cac so nguyen to trong a";
			//Bo sung sau
			break;

		case 5:
			system("CLS");
			cout << endl << "5. Tinh tong cac gia tri  trong mang";
			//Bo sung sau
			break;
		case 6:
			system("CLS");
			cout << endl << "6. Tính tong cac gia tri chi xuat hien mot lan trong mang";
			//Bo sung sau
			break;
		case 7:
			system("CLS");
			cout << endl << "7. Tinh tong cac gia tri phan biet trong mang";
			//Bo sung sau
			break;
	}
	_getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

3.4 Bổ sung khai báo nguyên mẫu các hàm tổ chức menu trong phần khai báo nguyên mẫu hàm

```
void XuatMenu();
int ChonMenu(int soMenu);
void XuLyMenu(int menu, DaySo a, int &n);
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có .

- Trong tập tin program.cpp cập nhật lại hàm ChayChuongTrinh để điều khiển chọn và thực hiện menu.

```
void ChayChuongTrinh()
{
	// Khai bao bien
	int menu,			// luu so thu tu menu duoc chon
		soMenu = 7;	// luu so luong chuc nang
	int  n = 0; //kich thuoc khi dung cua mang va gia tri khoi tao
	DaySo a;
	//khong nhap mang a trong ham nhu bai 1
	do
	{
		menu = ChonMenu(soMenu);
		XuLyMenu(menu, a, n);
	} while (menu > 0); //menu == 0 thi dung chuong trinh
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có .
Kiểm tra chức năng 0 - Thoát khỏi chương trình..

Bước 4 :

Trong bước 4 này, ta làm công việc sau :

- Trong ***program.cpp***, khai báo bổ sung các thư viện cần thiết.
- Trong tập tin ***thuvien.h*** , soạn thảo các hàm nhập, xuất dãy số
- Trong tập tin ***menu.h*** bổ sung xử lý chức năng nhập, xuất dữ liệu trong hàm ***XuLyMenu.***

- Trong tập tin *program.cpp* khai báo bổ sung các thư viện sau :

```
#include <time.h>
#include <stdlib.h>
```

- Trong tập tin *thuvien.h* bổ sung các hàm nhập xuất :

4.1 Hàm nhập tự động dữ liệu mảng 1 chiều.

```
// Input : a = mảng một chiều chứa tối đa MAX phần tử.
//         n = số phần tử thực sự được lưu trong mảng.
// Output: Không có.
void NhapTuDong(DaySo a, int n)
{
	srand((unsigned) time(NULL));
	for (int i = 0; i<n; i++)
	{
		a[i] = (MAX / 2 - rand() % MAX) / 2; //ngẫu nhiên trong khoảng [-8,8]
	}
}
```

4.2 Định nghĩa hàm xuất các phần tử của mảng ra màn hình

```
void XuatMang(DaySo a, int n)
{
	int i;
	for (i=0; i<n; i++)
		cout << a[i] << TAB;
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

4.3 Bổ sung nguyên mẫu các hàm :

```
void NhapTuDong(DaySo a, int n);
void XuatMang(DaySo a, int n);
```

- Trong tập tin *menu.h* ta bổ sung xử lý chức năng nhập,xem dữ liệu của mảng.

(Các case từ 3 đên 7 giữ nguyên, bổ sung xử lý trong case 1, case 2)

```
void XuLyMenu(int menu, DaySo a, int &n)
{
	// Khai báo biến
	switch (menu)
	{

		case 0:
			system("CLS");
			cout <<endl<< "\n0. Thoat khoi chuong trinh\n";
			break;
		case 1:
			system("CLS");
			cout << endl << "1. Nhap tu dong mang a";
			cout << "\nNhap kich thuoc n : ";
```

```
                cin >> n;
                //goi ham nhap du lieu
                NhapTuDong(a, n);

                system("CLS");
                cout << "\nDay so moi nhap:\n";
                XuatMang(a, n);
                break;

        case 2:
                cout << endl << "7. Xem du lieu day so";
                cout << "\nDay so hien hanh:\n";
                XuatMang(a, n);
                break;
        //. . .
    }
    _getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.
Kiểm tra thực hiện việc chọn chức năng 1 (nhập dữ liệu), chức năng 2 (xem dữ liệu).

Trong các bước tiếp theo, ta soạn thảo từng hàm chức năng trong tập tin ***thuvien.h***, bổ sung xử lý chức năng trong hàm ***XuLyMenu*** của ***menu.h***.

Bước 5: Bổ sung chức năng 3 (đếm số làn xuất hiện của x trong a) vào chương trình.
- Trong ***thuvien.h*** :
5.1 Định nghĩa hàm đếm số lần phần tử x xuất hiện trong a?

```
int Dem_X(DaySo a, int n, int x)
{
    int i, dem = 0;
    for (i = 0; i < n; i++)
    if (a[i] == x)
        dem++;
    return dem;
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

5.2 - Khai báo nguyên mẫu hàm :

```
int Dem_X(DaySo a, int n, int x);
```

- Trong ***menu.h*** :
Trong hàm ***XuLyMenu*** bổ sung khai báo biến x kiểu int (để lưu trử giá trị cần xét nhập vào từ bàn phím), biến kq kiểu int để lưu trử két quả tính toán, bổ sung việc thực hiện chức năng đếm x trong case 3, các case khác giữ nguyên.

```
void XuLyMenu(int menu, DaySo a, int &n)
{
    // Khai báo biến
    int x, kq;
    switch (menu)
    {
        //. . .
```

```
        case 3:
                system("CLS");
                cout << endl << "3. Dem so lan xuat hien cua x trong a";
                //Bo sung sau
                cout << "\nNhap gia tri can xet: x = ";
                cin >> x;
                kq = Dem_X(a, n, x);
                system("CLS");
                cout << "\nSo lan " << x << " xuat hien trong a: kq =  " << kq;
                cout << "\nXem lai mang hien hanh de kiem tra :\n";
                XuatMang(a, n);
                break;
                //. . .
        }
        _getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.
Kiểm tra kết quả thực hiện chức năng 3.

Bước 6: Bổ sung chức năng 4 (đếm và xuất các số nguyên tố) vào chương trình

- Trong ***program.cpp***:
  Bổ sung thư viejn ***<math.h>***

- Trong ***thuvien.h*** :

```
6.1 Định nghĩa hàm  một số nguyên có phải la số nguyên tố
Intput : x; //số nguyên
Output :
        1; néu x nguyen tố
        0; ngược lại
int KiemTra_NT(int x)
{
        int i, m,
                kq;
        if (x < 2)
                kq = 0;
        else
        {
                m = (int)sqrt((double)x);
                kq = 1;
                for (i = 2; i <= m;i++)
                if (x % i == 0)
                {
                        kq = 0;
                        break;
                }
        }
        return kq;
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

```
6.2 Định nghĩa hàm  đếm và xuất các số nguyên tố trong a
Intput a,//day so
        N; //so nguyen
```

```
Output : dem //so cac so nguyen to trong a
int Dem_NT(DaySo a, int n)
{
	int i, dem = 0;
	cout << "\nCac so nguyen to trong a:\n";
	for (i = 0; i < n;i++)
	if (KiemTra_NT(a[i]))
	{
		dem++;
		cout << a[i] << TAB;
	}
	return dem;
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

6.3 Khai báo nguyên mẫu hàm :

```
int Dem_NT(DaySo a, int n);
int KiemTra_NT(int x);
```

- Trong ***menu.h*** :

Bổ sung việc thực hiện chức năng 4 trong case 4, các case khác giữ nguyên.

Hàm ***XuLyMenu*** cập nhật lại như sau :

```
void XuLyMenu(int menu, DaySo a, int &n)
{
	// Khai báo biến
	int x, kq;
	switch (menu)
	{
		//. . .
	case 4:
		system("CLS");
		cout << endl << "4. Dem va xuat cac so nguyen to trong a";
		kq = Dem_NT(a, n);
		if (kq)
			cout << "\nSo luong cac so nguyen to trong a : kq = "<<kq;
		else
			cout << "\nKhong co so nguyen to nao trong a.";
		cout << "\nXem lai mang hien hanh de kiem tra :\n";
		XuatMang(a, n);
		break;
		//. . .
	}
	_getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

Kiểm tra kết quả thực hiện chức năng 4.

Bước 7: Bổ sung chức năng 5 (tính tổng mảng) vào chương trình.

- Trong ***thuvien.h*** :

7.1 Định nghĩa hàm tính tổng các phần tử chỉ xuất hiện 1 lần:

Input: a,n

Output: sum = tổng giá trị các phần tử trong mảng

```
int TinhTong(DaySo a, int n)
{
	int  i,
	    sum = 0;
	for (i = 0; i < n; i++)
		sum += a[i];
	return sum;
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

7.2 Khai báo nguyên mẫu hàm :

```
int TinhTong(DaySo a, int n);
```

- Trong ***menu.h*** :

Bổ sung xử lý chức năng 5 trong case 5, các case khác giữ nguyên.

```
void XuLyMenu(int menu, DaySo a, int &n)
{
	// Khai báo biến
	int x, kq;
	switch (menu)
	{
		//. . .
	case 5:
		system("CLS");
		cout << endl << "5. Tính tong cac phan tu trong mang";
		cout << "\nTong cac phan tu trong mang: sum = "
			<< TinhTong(a, n);
		cout << "\nXem lai mang hien hanh de kiem tra :\n";
		XuatMang(a, n);
		break;
		//. . .
	}
	_getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

Kiểm tra thực hiện việc chọn chức năng 5.

Bước 8: Bổ sung việc thực hiện chức năng 6 (tính tổng duy nhất) vào chương trình

- Trong ***thuvien.h*** :

8.1 Định nghĩa hàm tính tổng các giá trị chỉ xuất hiện 1 lần:

Input: a,n

Output: sum = tổng các giá trị chỉ xuất hiện 1 lần

```
int TinhTongDuyNhat(DaySo a, int n)
{
	int i,
		sum = 0;
	for (i = 0; i < n; i++)
	if (Dem_X(a, n, a[i])==1)
```

```
                sum += a[i];
        return sum;
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

8.2 Khai báo nguyên mẫu hàm :

```
int TinhTongDuyNhat(DaySo a, int n);
```

- Trong ***menu.h*** :

Bổ sung xử lý chức năng 6 trong case 6, các case khác giữ nguyên.

```
void XuLyMenu(int menu, DaySo a, int &n)
{
        // Khai báo biến
        int x, kq;
        switch (menu)
        {
                //. . .
        case 6:
                system("CLS");
                cout << endl << "6. Tính tong cac phan tu chi xuat hien mot lan trong mang";
                cout << "\nTong cac phan tu trong mang chi xuat hien 1 lan: sum = "
                        << TinhTongDuyNhat(a, n);
                cout << "\nXem lai mang hien hanh de kiem tra :\n";
                XuatMang(a, n);
                break;
                //. . .
        }
        _getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

Kiểm tra kết quả thực hiện chức năng 6.

```
//=========================================
```

Bước 9: Bổ sung việc thực hiện chức năng 7 vào chương trình

- Trong ***thuvien.h*** :

```
9.1 Định nghĩa hàm tính tổng giá trị phân biệt
Input: a,n
Output: sum = tổng các giá trị phân biệt
int TinhTong_PhanBiet(DaySo a, int n)
{
        DaySo b; //b luu tru cac gia tri phan biet cua a
        int i,//duyet a
                m, //kich thuoc cua b
                j,//duyet b
                dau,//dsanh dau de nhan dang a[i] da xuat hien trong b
                sum = 0;
        m = 0;
        for (i = 0; i < n; i++)
        {
                dau = 1;//a[i] chua co trong b
                for (j = 0; j < m && dau; j++)
```

```
                dau = dau && (a[i] != b[j]);
            if (dau) //a[i] chua co trong b
            {
                b[m++] = a[i];
                sum += a[i];
            }
        }
        return sum;
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

9.2 Khai báo nguyên mẫu hàm :

```
int TinhTong_PhanBiet(DaySo a, int n);
```

- Trong ***menu.h*** :

Bổ sung xử lý chức năng 7 trong case 7, các case khác giữ nguyên.

```
void XuLyMenu(int menu, DaySo a, int &n)
{
        // Khai báo biến
        int x, kq;
        switch (menu)
        {
            //. . .
        case 7:
            system("CLS");
            cout << endl << "7. Tinh tong cac phan tu phan biet trong mang";
            kq = TinhTong_PhanBiet(a, n);
            cout << "\nTong cac phan tu phan biet trong a: sum = " << kq;
            cout << "\nXem lai mang hien hanh de kiem tra :\n";
            XuatMang(a, n);
            break;
            //. . .
        }
        _getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.
Kiểm tra thực hiện việc chọn chức năng 7.
Kiểm tra cac chức năng chương trình. Kết thúc.

//

Bài 3: Minh họa các bài chèn, xóa, thay thế, sắp xếp

Viết chương trình thực hiện các thao tác trên dãy a gồm n số nguyên. Chương trình yêu cầu nhập dữ liệu cho a và cho phép người dùng chọn các chức năng trong menu:

- Nhập tự động dãy số.
- Xem dữ liệu dãy số
- Chèn giá trị x vào đầu dãy số
- Xóa phần tử cuối dãy
- Cắt phần tử đầu dãy rồi chèn vào cuối dãy
- Thay thế giá trị x trong dãy số bằng giá trị y
- Sắp dãy tăng dần

- Sắp dãy theo yêu cầu:
  - Đầu dãy là các số dương tăng dần
  - Tiếp theo là các số âm giảm dần
  - Cuối cùng là các số 0.

Bước 1. Tạo dự án Win32 Console Application mới. Đặt tên là **Lab05_D_Bai3**
Bước 2. Tạo cấu trúc cho chương trình như đã hướng dẫn trong **mục 2 lab 4** (từ 1- 8 để có cấu trúc nội dung tối thiểu chạy được chương trình).
Bước 3:

- Trong tập tin ***thuvien.h***, ta bổ sung định nghia hằng, kiểu dũ liệu mới :

```
//Dinh nghia hang, kieu du lieu moi
//Dinh nghia hang
#define MAX 100 //kich thuoc khai bao mang 1chieu
#define   TAB  '\t'

//Dinh nghia kieu du lieu moi:
typedef int DaySo[MAX];

//khai bao nguyen mau cac ham xu ly, nhap xuat


//Dinh ngia cac ham xu ly, nhap xuat
```

- Trong tập tin ***menu.h*** ta viết lại như sau (cấu trúc giống như bước 9 mục 2 lab 4, chỉ thay đổi nội dung theo yêu cầu bài toán) :

```
// Khai báo nguyên mẫu các hàm xử lý menu
        //bổ sung sau
// Định nghĩa các hàm xử lý menu

3.1 Định nghĩa hàm xuất danh sách chức năng ra màn hình
void XuatMenu()
{
        cout << endl << "============== CHON CHUC NANG =============";
        cout << endl << "0. Thoat khoi chuong trinh";
        cout << endl << "1. Nhap tu dong day  a";
        cout << endl << "2. Xem du lieu day a";
        cout << endl << "3. Chen x vao dau day";
        cout << endl << "4. Xoa phan tu dau day";
        cout << endl << "5. Cat phan tu dau day roi chen vao cuoi day";
        cout << endl << "6. Thay the cac gia tri x trong a bang gia tri y";
        cout << endl << "7. Sap day tang dan";
        cout << endl << "8. Sap day theo yeu cau :Duong Tang – Am Giam - Khong";
        cout << endl << "=========================================";
}

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.


3.2 Định nghĩa hàm chọn một menu trong danh sách
        Viết hàm ChonMenu như 3.2 bài 2.
Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.
```

3.3 Định nghĩa hàm xử lý menu :

Chú ý rằng các thao tác đều thực hiện trên cùng một đầu vào là mảng kiểu DaySo, nên ta bổ sung thêm biến mảng a kiểu DaySo với kích thước mảng là số nguyên dương n làm đối của hàm **XuLyMenu** (ngoài tham số menu). Để lưu giữ sự thay đổi dữ liệu trong a, n sẽ được viết dạng tham chiếu.

```
// Input : menu = Số thứ tự menu do người chọn,
                  Dãy số a
//                 số nguyên dương n
// Output: Không có.
void XuLyMenu(int menu, DaySo a, int &n)
{
        // Khai báo biến

        switch (menu)
        {
                case 0:
                        system("CLS");
                        cout << endl<<"\n0. Thoat khoi chuong trinh.\n";
                        break;
                case 1:
                        system("CLS");
                        cout << endl << "1. Nhap tu dong day a";
                        //Bo sung sau
                        break;
                case 2:
                        system("CLS");
                        cout << endl << "2. Xem du lieu day a";
                        //Bo sung sau
                        break;
                case 3:
                        system("CLS");
                        cout << endl << "3. Chen x vao dau day";
                        //Bo sung sau
                        break;
                case 4:
                        system("CLS");
                        cout << endl << "4. Xoa phan tu dau day";
                        //Bo sung sau
                        break;

                case 5:
                        system("CLS");
                        cout << endl << "5. Cat phan tu dau day roi chen vao cuoi day";
                        //Bo sung sau
                        break;
                case 6:
                        system("CLS");
                        cout << endl << "6. Thay the cac gia tri x trong a bang gia tri y";
                        //Bo sung sau
                        break;
                case 7:
                        system("CLS");
                        cout << endl << "7. Sap day tang dan";
```

```
                        //Bo sung sau
                        break;
                case 8:
                        system("CLS");
                        cout << endl << "8. Sap day theo yeu cau :Duong Tang – Am Giam - Khong";
                        //Bo sung sau
                        break;
        }
        _getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

3.4 Bổ sung khai báo nguyên mẫu các hàm tổ chức menu trong phần khai báo nguyên mẫu hàm

```
void XuatMenu();
int ChonMenu(int soMenu);
void XuLyMenu(int menu, DaySo a, int &n);
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có

- Trong tập tin program.cpp cập nhật lại hàm ChayChuongTrinh để điều khiển chọn và thực hiện menu.

```
void ChayChuongTrinh()
{
        // Khai bao bien
        int menu,                   // luu so thu tu menu duoc chon
                soMenu = 8;     // luu so luong chuc nang
        int  n = 0; //kich thuoc khi dung cua mang va gia tri khoi tao
        DaySo a;
        do
        {
                menu = ChonMenu(soMenu);
                XuLyMenu(menu, a, n);
        } while (menu > 0); //menu =0 thi dung chuong trinh
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có .
Kiểm tra kết quả thực hiện chức năng 0 (thoát khỏi chương trình).

Bước 4 :

Trong bước 4 này, ta làm công việc sau :

- Trong ***program.cpp***, khai báo bổ sung các thư viện cần thiết.
- Trong tập tin ***thuvien.h*** , soạn thảo các hàm nhập, xuất dãy số
- Trong tập tin ***menu.h*** bổ sung xử lý chức năng nhạp, xem dữ liệu trong hàm ***XuLyMenu.***

- Trong tập tin *program.cpp* khai báo bổ sung các thư viện sau :

```
#include <time.h>
#include <stdlib.h>
```

- Trong tập tin *thuvien.h* bổ sung các hàm nhập xuất :

4.1 Hàm nhập tự động dữ liệu mảng 1 chiều.

Ta viết khác với bước 4 bài 2 một chút, đó là nhập kích thước mảng trong hàm, khi đó đối n viết dưới dạng tham chiếu.

```
void NhapTuDong(DaySo a, int  &n)
```

```
{
	int  i;
	cout << "\nNhap kich thuoc n : ";
	cin >> n;
	srand((unsigned) time(NULL));
	for ( i = 0; i<n; i++)
		a[i] = (MAX / 2 - rand() % MAX) / 6;
}
```

4.2 Định nghĩa hàm xuất các phần tử của mảng ra màn hình

```
void XuatMang(DaySo a, int n)
{
	int i;
	for (i=0; i<n; i++)
		cout << a[i] << TAB;
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có

4.3 Bổ sung nguyên mẫu các hàm :

```
void NhapTuDong(DaySo a, int  n);
void XuatMang(DaySo a, int n);
```

- Trong tập tin *menu.h* ta bổ sung xử lý chức năng nhập,xem dữ liệu của mảng.

(Các case từ 3 đên 7 giữ nguyên, bổ sung xử lý nhập, xuất trong case 1, case 2)

```
void XuLyMenu(int menu, DaySo a, int &n)
{
	// Khai báo biến
	switch (menu)
	{

		case 0:
			system("CLS");
			cout <<endl<< "\n0. Thoat khoi chuong trinh\n";
			break;
		case 1:
			system("CLS");
			cout << endl << "1. Nhap tu dong mang a";
			//goi ham nhap du lieu
			NhapTuDong(a, n);

			system("CLS");
			cout << "\nDay so moi nhap:\n";
			XuatMang(a, n);
			break;

		case 2:
			cout << endl << "7. Xem du lieu day so";
			cout << "\nDay so hien hanh:\n";
			XuatMang(a, n);
			break;
		//. . .
	}
	_getch();
```

```
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.
Kiểm tra thực hiện việc chọn chức năng 1 (nhập dữ liệu), chức năng 2 (xem dữ liệu).

Trong các bước tiếp theo, ta soạn thảo từng hàm chức năng trong tập tin ***thuvien.h***, bổ sung xử lý chức năng vào hàm ***XuLyMenu*** trong ***menu.h***

Bước 5: Bổ sung việc thực hiện chức năng 3 (chèn vào đầu dãy) vào chương trình.
- Trong ***thuvien.h*** :

```
5.1 Định nghĩa hàm chèn x vào đầu dãy a?
//Input : Day a, kich thuoc  n, gia tri x can chen
//Output : Day a(them x o dau)

void ChenDauDay(DaySo a, int &n, int x)
{
	int i;
	for (i = n - 1; i >= 0; i--)
		a[i + 1] = a[i];//doi ra sau 1 vi tri, bat dau tu cuoi mang
	a[0] = x;//gan x tai vi tri dau mang
	n++;//kich thuoc mang tang len 1
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

```
5.2 - Khai báo nguyên mẫu hàm :
void ChenDauDay(DaySo a, int &n, int x);
```

- Trong ***menu.h*** :

Trong hàm XuLyMenu ta bổ sung khai báo biến x kiểu int (để lưu trữ giá trị cần chèn nhập vào từ bàn phím), bổ sung xử lý chức năng chèn x vào đầu dãy trong case 3, các case khác giữ nguyên.

```
void XuLyMenu(int menu, DaySo a, int &n)
{
	// Khai báo biến
	int x;
	switch (menu)
	{
	//. . .
	case 3:
		system("CLS");
		cout << endl << "3. Chen x vao dau day";
		cout << "\nNhap gia tri can chen: x = ";
		cin >> x;
		cout << "\nKich thuoc mang hien hanh : n = " << n;
		cout << "\nDay so hien hanh:\n";
		XuatMang(a, n);
		ChenDauDay(a, n, x);
		cout << "\n\nKich thuoc mang ket qua : n = " << n;
		cout << "\nDay so ket qua sau khi chen "<<x<<" vao dau day :\n";
		XuatMang(a, n);
		break;
	//. . .
	}
```

```
	_getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.
Kiểm tra kết quả thực hiện chức năng 3.

Bước 6: Bổ sung chức năng 4 (xóa giá trị đầu dãy) vào chương trình.
- Trong ***thuvien.h*** :

```
6.1 Định nghĩa hàm
//Ham xoa gia tri dau day
//Input : Day a, kich thuoc n,
//Output : Day a(bot vi tri dau day)

void XoaDauDay(DaySo a, int &n)
{
	int i;
	for (i = 1; i < n; i++)
		a[i - 1] = a[i];//doi ve truoc 1 vi tri, bat dau tu vi tri 1
	n--;//kich thuoc mang giam bot 1
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

6.2 - Khai báo nguyên mẫu hàm :

```
void XoaDauDay(DaySo a, int &n);
```

- Trong ***menu.h*** :

Nội dung hàm XuLyMenu bổ sung việc thực hiện chức năng 4 vào case 4, các case khác giữ nguyên.

```
void XuLyMenu(int menu, DaySo a, int &n)
{
	// Khai báo biến
	int x;
	switch (menu)
	{
	//. . .
	case 4:
		system("CLS");
		cout << endl << "4. Xoa phan tu cuoi day";
		cout << "\nKich thuoc mang hien hanh : n = " << n;
		cout << "\nDay so hien hanh:\n";
		XuatMang(a, n);
		XoaDauDay(a, n);
		cout << "\n\nKich thuoc mang ket qua : n = " << n;
		cout << "\nDay so ket qua sau khi xoa gia tri dau :\n";
		XuatMang(a, n);
		break;
	//. . .
	}
	_getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.
Kiểm tra thực hiện việc chọn chức năng 4.

Bước 7: Bổ sung việc thực hiện chức năng 5 (Cắt phần tử đầu rồi chèn vào cuối dãy) vào chương trình

- Trong ***thuvien.h*** :

7.1 Định nghĩa hàm cắt phần tử đầu rồi chèn cuối dãy

```
//Ham cat dau chen cuoi
void CatDau_ChenCuoi(DaySo a, int &n)
{
	int  i,
	     x; //luu phan tu dau
	x = a[0];
	for (i = 1; i < n; i++)//Xoa dau
		a[i - 1] = a[i];//doi ve truoc 1 vi tri, bat dau tu vi tri 1
	a[n-1] = x;//gan x tai vi tri cuoi mang
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

7.2 - Khai báo nguyên mẫu hàm :

```
void CatDau_ChenCuoi(DaySo a, int &n);
```

- Trong ***menu.h*** :

Trong hàm XuLyMenu bổ sung xử lý chức năng 5 vào case 5, các case khác giữ nguyên.

```
void XuLyMenu(int menu, DaySo a, int &n)
{
	// Khai báo biến
	int x;
	switch (menu)
	{
	//. . .
	case 5:
		system("CLS");
		cout << endl << "5. Cat phan tu dau day roi chen vao cuoi day";
		cout << "\nKich thuoc mang hien hanh : n = " << n;
		cout << "\nDay so hien hanh:\n";
		XuatMang(a, n);
		CatDau_ChenCuoi(a, n);
		cout << "\n\nKich thuoc mang ket qua : n = " << n;
		cout << "\nDay so ket qua sau khi cat gia tri dau roi chen vao cuoi day:\n";
		XuatMang(a, n);
	//. . .
	}
	_getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

Kiểm tra thực hiện việc chức năng 5.

Bước 8: Bổ sung việc thực hiện chức năng 6 (thay thế x bằng y) vào chương trình

- Trong ***thuvien.h*** :

8.1 Định nghĩa hàm thay the x trong day bang y

```
//Input : Day a, kich thuoc n, gia tri x can thay the, gia tri thay the y
//Output : Day a(x thay boi y)

void Thay_X_Bang_Y(DaySo a, int &n, int x, int y)
{
	int i;
```

```
	for (i = 0; i < n; i++)
	if (a[i] == x)
		a[i] = y;
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

8.2 - Khai báo nguyên mẫu hàm :

```
void Thay_X_Bang_Y(DaySo a, int &n, int x, int y);
```

- Trong ***menu.h*** :

Nội dung hàm XuLyMenu bổ sung thêm khai báo biến y kiểu int để lưu trử giá trị thay thế, biến x đã có sẽ lưu trử giá trị cần thay thế, bổ sung nội dung xử lý chức năng 6 vào case 6, các case khác giữ nguyên.

```
void XuLyMenu(int menu, DaySo a, int &n)
{
	// Khai báo biến
	int x, y;
	switch (menu)
	{
	//. . .
	case 6:
		system("CLS");
		cout << endl << "6. Thay the gia tri x trong a bang gia tri y";
		cout << "\nNhap gia tri can thay the: x = ";
		cin >> x;
		cout << "\nNhap gia tri  thay the: y = ";
		cin >> y;

		cout << "\nKich thuoc mang hien hanh : n = " << n;
		cout << "\nDay so hien hanh:\n";
		XuatMang(a, n);
		Thay_X_Bang_Y(a, n, x,y);
		cout << "\n\nKich thuoc mang ket qua : n = " << n;
		cout << "\nDay so ket qua sau khi thay " << x << " bang gia tri "<<y<<":\n";
		XuatMang(a, n);
		break;

	//. . .
	}
	_getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

Kiểm tra thực hiện việc chức năng 6.

Bước 9: Bổ sung chức năng 7 (sắp tăng dãy) vào chương trình

- Trong ***thuvien.h*** :

9.1 Định nghĩa hàm sắp tăng dãy:

```
//Input : Day a, kich thuoc n
//Output : Day a(da tang)
void SapTang(DaySo a, int n)
{
	int i, j;
	for (i = 0; i < n - 1; i++)
```

```
	for (j = i + 1; j < n; j++)
	if (a[i]>a[j])
		HoanVi(a[i], a[j]);
}
```

9.2 Định nghĩa hàm hoán vị:

```
void HoanVi(int &x, int &y)
{
	int tam;
	tam = x;
	x = y;
	y = tam;
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

9.3 - Khai báo nguyên mẫu hàm :

```
void SapTang(DaySo a, int n);
void HoanVi(int &x, int &y);
```

- Trong ***menu.h*** :

Nội dung hàm XuLyMenu bổ sung nội dung thực hiện chức năng 7 vào case 7, các case khác giữ nguyên.

```
void XuLyMenu(int menu, DaySo a, int &n)
{
	// Khai báo biến
	int x, y;
	switch (menu)
	{
	//. . .
	case 7:
		system("CLS");
		cout << endl << "7. Sap day tang dan";
		cout << "\nDay so hien hanh:\n";
		XuatMang(a, n);
		SapTang(a, n);
		cout << "\nDay so sau khi sap tang:\n";
		XuatMang(a, n);
		break;

	//. . .
	}
	_getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

Kiểm tra thực hiện việc chức năng 7.

Bước 10: Bổ sung chức năng 8 (sắp dãy theo yêu cầu) vào chương trình.

- Trong ***thuvien.h*** :

10.1 Định nghĩa hàm sắp dãy theo yêu cầu : Dương tăng-Âm giảm-Không

```
//Input : Day a, kich thuoc n
//Output : Day a(da theo yeu cau)
void Sap_DuongTang_AmGiam_Khong(DaySo a, int n)
{
	int i, j, mc;
```

```
	for (i = 0; i < n - 1; i++)
	for (j = i + 1; j < n; j++)
	{
		mc = (a[i] < 0 && a[j] < 0 && a[i] < a[j]) ||
			(a[i] < 0 && a[j]>0 )||
			(a[i] == 0 && a[j] != 0) ||
			(a[i] > 0 && a[j] > 0 && a[i] > a[j]);
		if (mc)
			HoanVi(a[i], a[j]);
	}
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

10.2 - Khai báo nguyên mẫu hàm :

```
void Sap_DuongTang_AmGiam_Khong(DaySo a, int n);
```

- Trong ***menu.h*** :

Trong hàm XuLyMenu, bổ sung xử lý chức năng 8 vào case 8, các case khác giữ nguyên.

```
void XuLyMenu(int menu, DaySo a, int &n)
{
	// Khai báo biến
	int x, y;
	switch (menu)
	{
	//. . .
	case 8:
		system("CLS");
		cout << endl << "8. Sap day theo yeu cau :Duong Tang - Am Giam - Khong";
		cout << "\nDay so hien hanh:\n";
		XuatMang(a, n);
		Sap_DuongTang_AmGiam_Khong(a, n);
		cout << "\nDay so sau khi sap theo yeu cau:\n";
		XuatMang(a, n);
		break;
	//. . .
	}
	_getch();
}
```

Nhấn Ctrl+F5 để chạy chương trình, sửa lỗi nếu có.

Kiểm tra chức năng 8.

Kiểm tra tất cả các chức năng. Kết thúc chương trình.

Bài 4:

Viết chương trình nhập một dãy n số nguyên, xuất ra các giá trị phân biệt của dãy và số lần xuất hiện của nó trong dãy.

Chương trình này tổ chức như mục 1, phần C, lab 4 : chỉ có 2 tập tin : program.cpp và thuvien.h, không có tập tin menu.h vì bài toán không yêu cầu tổ chức tùy chọ menu

Bước 1. Tạo dự án Win32 Console Application mới. Đặt tên là **Lab05_D_Bai4**

Bước 2. Tạo cấu trúc chương trình như đã hướng dẫn trong **mục 1 phần C lab 4** (từ bước 1 đến bước 7).

Bước 3.

- Trong ***program.cpp*** ta khai báo bổ sung :

```
#include <time.h>
```

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

- Trong tập tin ***thuvien.h***, từng bước bổ sung định nghĩa hằng, cài đặt các hàm xử lý, . . . :

```
//Định nghĩa hằng

 #define MAX 100
#define TAB '\t'

//Khai báo nguyên mẫu các hàm xử lý

//. . . (bổ sung sau)
```

//Định nghĩa các hàm xử lý

3.1 Hàm nhập tự động mảng n số nguyên

```
void NhapTuDong(int a[MAX], int &n)
{
	int i;
	cout << "\nNhap kich thuoc n : ";
	cin >> n;
	srand((unsigned)time(NULL));
	for (i = 0; i<n; i++)
		a[i] = (MAX / 2 - rand() % MAX) / 6; //Trong khoang [-8,+8]
}
```

Nhấn Ctrl+F5 chạy chương trình, sửa lỗi nếu có.

3.2 Hàm xuất mảng n số nguyên

```
void XuatMang(int a[MAX], int n)
{
	int i;
	for (i = 0; i<n; i++)
		cout << a[i] << TAB;
}
```

Nhấn Ctrl+F5 chạy chương trình, sửa lỗi nếu có.

3.3 Hàm tìm các giá trị phân biệt của a và số lần xuất hiện của nó.

```
//Input : a, n
//Output: b, ( cac gia tri phan biet cua a luu trong mang b)
//		c, (so lần xuat hien cua b[j] luu trong c[j])
//		m ( kich thuoc cua b,c )
//b,c,m  lam doi ra cua ham

void Tim_Day_GiaTri_PhanBiet(int a[MAX], int n, int b[MAX], int c[MAX], int &m)
{
	int i; //duyet theo n
	int j; //duyet b,c
	int dau; //danh dau  a[i] co thuoc b
	for (i = 0; i < n; i++)
		c[i] = 1; //khoi tao c : so lan xuat hiem cua moi gia tri phan bat b[i] bang 1
	m = 0; //khoi tao kich thuoc b, c
	for (i = 0; i < n; i++)
	{
```

```
                dau = 0; //a[i] khong thuoc b
                for (j = 0; j < m; j++)
                if (a[i] == b[j])
                {
                        dau = 1;//a[i] thuoc b
                        c[j]++;//b[j]  tang them 1 lan xuat hien
                        break;
                }
                if (!dau)//khong thuoc
                {
                        b[m] = a[i]; // chen a[i] vao cuoi b
                        m++;
                }
        }
}
```

(Xem thêmtại trang 24 cách viết hàm : int TinhTong_PhanBiet(DaySo a, int n))

Nhấn Ctrl+F5 chạy chương trình, sửa lỗi nếu có.

3.4 Khai báo nguyên mẫu các hàm :

```
void XuatMang(int a[MAX], int n);
void NhapTuDong(int a[MAX], int &n);
void Tim_Day_GiaTri_PhanBiet(int a[MAX], int n, int b[MAX], int c[MAX], int &m);
```

Nhấn Ctrl+F5 chạy chương trình, sửa lỗi nếu có.

Bước 4: Tích hợp các chức năng để hoàn chỉnh chương trình

Trong tập tin ***program.cpp :***

- Khai báo bổ sung thư viện :

```
#include <iomanip>
```

- Bổ sung hàm ChayChuongTrinh với các nội dung :

Nhập tự động dữ liệu cho mảng, gọi hàm tìm giá trị phân biệt của a và số lần xuất hiên, xuất kết quả.Có thể điều khiển lặp việc chương trình cho đến khi người dùng dừng.

```
void ChayChuongTrinh()
{
        Char  kt;
        int  a[MAX], b[MAX], c[MAX];
        int   i, n = 0, m = 0;
        do
        {
                system("CLS");
                NhapTuDong(a, n);
                Tim_Day_GiaTri_PhanBiet(a, n, b, c, m);
                cout << "\nDay dang xet:\n";
                XuatMang(a, n);
                cout << setiosflags(ios::left);
                cout << endl << setw(20) << "Gia tri Phan Biet"
                        << setw(20) << "So lan xuat hien";
                for (i = 0;  i < m; i++)
                {
                        cout << endl << setw(20) << b[i]
```

```
                        << setw(20) <<c[i];
                }
                cout << "\nNua khong, nhan ESC neu khong!\n";
                kt = _getch();
        } while (kt != 27);
}
```

Nhấn Ctrl+F5 chạy chương trình, sửa lỗi nếu có.
Thực hiện chương trình – kết thúc chương trình.

E. Bài tập bắt buộc

1. Tìm kiếm

- Tìm vị trí của số nguyên tố cuối cùng trong mảng ***a***. Nếu ***a*** không chứa số nguyên tố, trả về -1.
- Tìm phần tử xuất hiện nhiều nhất và số lần xuất hiện của nó.
- Tìm phần tử có giá trị nhỏ nhất trong mảng và vị trí xuất hiện đầu tiên của nó.
- Tìm số âm lớn nhất và vị trí của nó.
- Tìm số dương nhỏ nhất và vị trí của nó.

2. Đếm

- Đếm số lượng số có 3 chữ số.
- Đếm các số nằm ngoài phạm vi [***min*** .. ***max***] cho trước.
- Đếm số lượng số chính phương (số chính phương là số bằng bình phương 1 số khác. Ví dụ: 4, 9, 16, 25, ...)
- Đếm số lần xuất hiện của phần tử ***x*** kể từ vị trí ***vt*** cho trước.
- Đếm số lượng các đường chạy trong dãy. Biết rằng, đường chạy là dãy con có thứ tự (tăng hoặc giảm) dài nhất gồm những phần tử nằm kế tiếp nhau.

3. Sắp xếp

- Sắp các số dương tăng dần, các số khác giữa nguyên vị trí
- Sắp các phần tử sao cho số 0 nằm ở cuối mảng, các số khác ở đầu mảng và tăng dần.
- Sắp các phần tử sao cho số 0 ở đầu mảng, số âm ở giữa và giảm dần, số dương ở cuối và tăng
- Sắp các số lẻ nằm đầu mảng và tăng dần, các số chẵn nằm cuối mảng và giảm dần.
- Sắp các số nguyên tố nằm đầu mảng và tăng, các số còn lại nằm ở cuối và giảm dần.

4. Chèn và thay thế

- Chèn phần tử ***x*** vào mảng ***a*** tại vị trí ***vt*** cho trước.
- Chèn phần tử ***x*** vào sau phần tử lớn nhất (đầu tiên tìm được) trong mảng.
- Chèn phần tử ***x*** vào trước số nguyên tố đầu tiên trong mảng.
- Chèn phần tử ***x*** vào sau mỗi phần tử ***y*** cho trước. Nếu mảng không chứa ***y*** thì chèn tại vị trí 0.
- Thay thế giá trị nhỏ nhất bằng giá trị ***x*** cho trước.

5. Xóa

- Xóa phần tử nằm tại vị trí ***vt*** cho trước khỏi mảng ***a***.

- Xóa phần tử ***x*** đầu tiên tìm được trong mảng ***a***.
- Xóa mọi phần tử ***x*** trong mảng ***a***.
- Xóa tất cả các phần tử trùng nhau, chỉ giữ lại một phần tử trong số các phần tử trùng đó.
- Xóa các phần tử nằm ngoài đoạn [***min*** .. ***max***] cho trước.

6. Tính toán

- Tính trung bình cộng của các phần tử trong mảng
- Tính tổng bình phương của các phần tử trong mảng
- Tính độ lệch lớn nhất giữa 2 phần tử nằm liên tiếp nhau
- Tính tổng các số nguyên tố có 2 chữ số

7. Kiểm tra đúng sai của các phát biểu sau

- Mảng a không chứa phần tử 0.
- Mảng a có chứa 3 phần tử nằm liên tiếp có giá trị liên tiếp nhau.
- Mảng a chứa cả phần tử 0 lẫn 1.
- Mảng a chứa phần tử có giá trị bằng trung bình cộng của các phần tử.
- Mảng a không chứa giá trị âm.

F. Bài tập làm thêm

1. Thống kê

Cho mảng a chứa các số nguyên trong đoạn [0..10000]. Viết chương trình xuất ra màn hình các phần tử phân biệt của mảng a và số lần xuất hiện của chúng.

2. Bài cào

Bài cào là một kiểu chơi bài bằng bộ bài tây 52 lá. Bài được chia cho từng người, mỗi người 3 lá. Điểm của người chơi trong mỗi ván là số lẻ của tổng điểm 3 lá bài. Ví dụ, nếu tổng điểm 3 lá bài là 27 thì người đó được 7 điểm, nếu tổng là 10 điểm thì được 0 điểm. Cách tính điểm của các lá bài như sau:

- Các lá 2, 3, ..., 10 mỗi lá có điểm tương ứng với con số đó, bất kể lá bài màu gì.
- Lá A có điểm là 1, các lá J, Q, K đều được tính là 10 điểm.

Sau khi tính điểm và trình bài, ai có số điểm cao nhất là thắng ván đó. Trường hợp đặc biệt, ai sở hữu được cả 3 lá bài đều là bài tây (J, Q hoặc K) hoặc cả 3 lá đều cùng điểm số thì thắng ngay ván đó, không cần tính điểm. Nếu có từ 2 người trở lên có cùng điểm số cao nhất thì tiền cược được chia đều.

Viết chương trình minh họa trò chơi theo mô tả trên. Trong đó, máy tính đóng vai trò người chia bài. Sau mỗi ván, máy phải xáo bài trước khi chia. Có tất cả 10 người chơi, mỗi người được cấp một số tiền M. Chương trình sẽ dừng khi chỉ còn 1 người đủ tiền đặt cược hoặc khi người dùng chọn chức năng thoát chương trình. Tiền cược quy định cho mỗi ván là C với C <= M/10. Những người chơi có số tiền bé hơn C không được phép tham gia tiếp.

Chương trình phải có các chức năng sau: thiết lập mức cược C, chia bài, tính điểm và thoát chương trình.

3. Xếp hạng

Cho mảng a chứa tối đa 10000 số nguyên phân biệt. Hãy viết chương trình xuất ra màn hình lần lượt từng phần tử của a và thứ tự của nó trong dãy a sau khi đã sắp xếp.

4. Tìm bia

Một nhóm bạn tham dự một bữa tiệc. Mỗi người được phép uống 1 lon bia. Trong lúc đang ăn uống thì một cố xảy ra gây tắt đèn và chuông báo động cháy nổ vang lên. Mọi người bình tĩnh đặt lon bia xuống bàn rồi thoát ra

khỏi tòa nhà. Sau khi báo động tắt, sự cố được giải quyết, họ trở lại bữa tiệc và cố tìm lại lon bia của mình. Tuy nhiên, đa số họ quên mất trước đó mình ở vị trí nào. Do đó, họ cứ lấy ngẫu nhiên 1 lon bia. Tính xác suất để có ít nhất một người lấy đúng lon bia của mình trước đó.

Gợi ý:

- *Viết một chương trình cho phép nhập vào số lượng sinh viên (N) tham dự bữa tiệc và giả lập M lần (M >= 1000) sự kiện được mô tả trong đề bài.*
- *Giả sử ban đầu các lon bia được đánh số từ 1 tới N ứng với số thứ tự từng sinh viên.*
- *Sử dụng phương pháp sinh ngẫu nhiên N số phân biệt từ 1 tới N hoặc xáo trộn ngẫu nhiên N số ban đầu để giả lập sự kiện lấy lại bia.*
- *Tính số thí nghiệm (gọi là F) xảy ra trường hợp có ít nhất 1 sinh viên lấy đúng bia của mình.*
- *In ra màn hình giá trị F/M.*
- *Thử thay đổi giá trị của N và M để đưa ra nhận xét.*

5. Phát hiện trùng lặp

Cho mảng a chứa N số nguyên có giá trị trong đoạn [1..N]. Hãy viết chương trình kiểm tra mảng có chứa ít nhất 2 phần tử trùng nhau hay không? Yêu cầu: chỉ được duyệt qua các phần tử của mảng một lần và không được dùng thêm mảng khác.

6. Nối dãy

Cho mảng a, b chứa tối đa MAX số thực. Hãy viết chương trình:

- Chèn các phần tử của mảng b vào cuối mảng a.
- Giả sử dãy a đã được sắp xếp tăng. Hãy chèn các phần tử của b vào a sao cho vẫn được thứ tự tăng.