Bài 1. Viết chương trình nhập một ma trận từ bàn phím sau đó ghi ma trận vào *file* theo cấu trúc sau: dòng đầu tiên có hai số nguyên mô tả số (`dòng`, `cột`) của ma trận tương ứng với hai giá trị (`m`, `n`); `m` dòng tiếp theo mỗi dòng có `n` số thực tương ứng với giá trị của từng phần tử.

1a. Viết hàm nhập ma trận

Bước	Ngôn ngữ tự nhiên	Ngôn ngữ C
1	Khai báo hàm có tên *nhapmatran* có 3 tham số: ma trận, số dòng, số cột của ma trận	void nhapmatran(int matran[20][20], **int *m, int *n**) {
2	Nhập số dòng của ma trận	Printf("Nhap so dong cua ma tran"); scanf("%d",m);
3	Nhập số cột của ma trận	Printf("Nhap so cot cua ma tran"); scanf("%d",n);

3	Nhập các phần tử của ma trận	Printf("Nhap cac phan tu cua ma tran") ;
		for (int i = 0; i < *m; i++)
		{
		for (int j = 0; j < *n; j++)
		{
		printf("Nhap matran[%d][%d]: ", i, j);
		scanf("%f", &matran[i][j]);
		}
		}

4	Kết thúc hàm	}

1b. Viết hàm ghi ma trận vào tệp

Bước	Ngôn ngữ tự nhiên	Ngôn ngữ C
1	Khai báo hàm có tên *ghimatranvaotep* có 4 tham số: ma trận, số dòng, số cột của ma trận và con trỏ có tên **tentep**	void ghimatranvaotep(int matran[20][20], int m, int n, **char *tentep** {
2	Khai báo biến con trỏ f thuộc kiểu tệp	FILE *f ;
3	Mở tệp có tên **teptin** để ghi ma trận vào gán cho biến f	f=fopen(tentep,"w");

4	Nhập các phần tử của ma trận
	
	Xuống dòng
		fprintf(f, "%d %d\n", m, n);
		
		for (int i = 0; i < m; i++) {
		for (int j = 0; j < n; j++) {
		fprintf(f, "%d ", matran[i][j]);
		}
		fprintf(f, "\n");
		}

5	Đóng tệp	fclose(f); }
6	Kết thúc hàm	}

Bài 2. Viết chương trình đọc nội dung của *file* chứa ma trận trên, sau đó in nội dung ma trận ra màn hình.

Bước	Ngôn ngữ tự nhiên	Ngôn ngữ C
1	Khai báo hàm có tên *docmatrantutep* có 4 tham số: ma trận, số dòng, số cột của ma trận và con trỏ có tên **tentep**	void docmatrantutep(int matran[20][20], int *m, int *n, **char *tentep** {
2	Khai báo biến con trỏ f thuộc kiểu tệp	FILE *f ;
3	Mở tệp có tên **teptin** để đọc ma trận vào gán cho biến f	f=fopen(tentep,"r");

4	Nhập các phần tử của ma trận
	
	Đọc các phần tử của ma trận từ tệp
	
	Đọc phần tử thứ i,j của ma trận từ tệp f
	In phàn tử thứ i,j của ma trận ra màn hình
	Viết hết 1 dòng thì xuống dòng viết dòng tiếp theo
		fscanf(f, "%d %d", m, n);
		
		printf("Ma tran doc tu tep:\n");
		
		for (int i = 0; i < *m; i++) {
		for (int j = 0; j < *n; j++) {
		fscanf(f, "%d", &matran[i][j]);// đọc
		printf("%d ", matran[i][j]);
		}
		printf("\n");
		}

5	Đóng tệp	fclose(f); }
6	Kết thúc hàm	}

Hàm main()

Bước	Ngôn ngữ tự nhiên	Ngôn ngữ C

		main() {
		int matran[20][20];
		int m, n;
		char tentep[] = "matran.txt";
		nhapmatran(matran, &m, &n);
		ghimatranvaotep(matran, m, n, tentep);
		docmatrantutep(matran, &m, &n, tentep);
		getch();
		}

Bài 4. Viết chương trình tạo một dãy số ngẫu nhiên gồm `n` số nguyên.

Bước	Ngôn ngữ tự nhiên	Ngôn ngữ C
1	Khai báo hàm có tên *taodaysongaunhien* có 2 tham số: con trỏ đại diện cho dãy số và 1 số nguyên n (số phần tử trong dãy)	void taodaysongaunhien(int *dayso, int n) {
2	Các phần tử của dãy số được tạo ngẫu nhiên bởi hàm **rand()**	for (int i = 0; i < n; i++) { dayso[i] = rand(); }
4	Kết thúc hàm	}

Ghi dãy số này vào *file* có tên là `dayso.dat`.

Bước	Ngôn ngữ tự nhiên	Ngôn ngữ C
1	Khai báo hàm có tên *ghidaysovaotep* có 3 tham số: dãy số, số phần tử của dãy số và con trỏ có tên **filename**	void ghidaysovaotep(int *dayso, int n, const char *filename) {
2	Khai báo biến con trỏ f thuộc kiểu tệp	FILE *f ;
3	Mở tệp có tên **filename** để ghi dãy số vào gán cho biến f	f=fopen(filename,"w");

4	Kiểm tra điều kiện nếu tệp rỗng thì không thể mở file ghi dữ liệu
	
	ghi giá trị n vào tệp f
	
	ghi lần lượt các giá trị của dãy số vào tệp
		if (file == NULL) {
		printf("khong the mo file ghi du lieu", filename);
		return;
		}
		fprintf(f, "%d\n", n);
		for (int i = 0; i < n; i++) {
		fprintf(f, "%d ", dayso[i]);
		}

5	Đóng tệp	fclose(f); }
6	Kết thúc hàm	}

Đọc và hiển thị nội dung của dãy từ *file* ra màn hình.

Bước	Ngôn ngữ tự nhiên	Ngôn ngữ C
1	Khai báo hàm có tên *docdaysotutep* có 3 tham số: dãy số, số phần tử của dãy số và con trỏ có tên **filename**	void docdaysotutep(int *dayso, int n, const char *filename) {
2	**Khai báo biến con trỏ f thuộc kiểu tệp**	FILE *f ;
3	Mở tệp có tên **filename** để ghi dãy số vào gán cho biến f	f=**fopen**(filename,"r");
4	Kiểm tra điều kiện nếu tệp rỗng thì không thể mở file ghi dữ liệu	if (file == NULL) { printf("khong the mo file ghi du lieu", filename); return;

	Đọc giá trị n vào tệp f
	
	ghi lần lượt các giá trị của dãy số vào tệp
		}
		fscanf(file, "%d", n);
		for (int i = 0; i < *n; i++) {
		fscanf(file, "%d", &dayso[i]);
		}
		
		fclose(file);
		}

5	Đóng tệp	fclose(f); }
6	Kết thúc hàm	}

Sắp xếp dãy số này theo thứ tự tăng dần.

Bước	**Ngôn ngữ tự nhiên**	**Ngôn ngữ C**
1	Khai báo hàm có tên *sapxepdaysotang* có 2 tham số: dãy số, số phần tử của dãy số	void sapxepdaysotang(int *dayso, int n) {
2	Duyệt từ phần tử thứ 0 đến phần tử thứ (n-2)	for (int i = 0; i < n - 1; i++) {
3	Duyệt từ phần tử thứ i+1 đến phần tử thứ (n-1)	for (int j = i+1; i < n ; j++) {

4	Kiểm tra điều kiện nếu phần tử thử i của dãy số lớn hơn phần tử thứ i+1 thì đổi vị trí của chúng	if (dayso[i] > dayso[j]) {
		int tg=dayso[i];
		dayso[i]=dayso[j];
		dayso[j]=tg;
		}
		}

5	Kết thúc hàm	}

Sắp xếp dãy số này theo thứ tự giảm dần.

Bước	**Ngôn ngữ tự nhiên**	**Ngôn ngữ C**
1	Khai báo hàm có tên *sapxepdaysogiam* có 2 tham số: dãy số, số phần tử của dãy số	void sapxepdaysogiam(int *dayso, int n) {
2	Duyệt từ phần tử thứ 0 đến phần tử thứ (n-2)	for (int i = 0; i < n - 1; i++) {
3	Duyệt từ phần tử thứ i+1 đến phần tử thứ (n-1)	for (int j = i+1; i < n ; j++) {

4	Kiểm tra điều kiện nếu phần tử thử i của dãy số lớn hơn phần tử thứ i+1 thì đổi vị trí của chúng	if (dayso[i] < dayso[j]) {
		int tg=dayso[i];
		dayso[i]=dayso[j];

		dayso[j]=tg;
		}
		}
		}

5	Kết thúc hàm	}

Hàm in dãy số

Bước	Ngôn ngữ tự nhiên	Ngôn ngữ C
1	Khai báo hàm có tên *indayso* có 2 tham số: dãy số, số phần tử của dãy số	void indayso(int *dayso, int n) {
2	Duyệt từ phần tử thứ 0 đến phần tử thứ (n-1)	for (int i = 0; i < n - 1; i++) {

4	In phần tử thứ i của dãy số	printf("%3d ", dayso[i]);
		}
		printf("\n");
		}

5	Kết thúc hàm	}

Hàm main()

Bước	Ngôn ngữ tự nhiên	Ngôn ngữ C

		main() {
		int n;
		printf("Nhap so luong phan tu cua day: ");
		scanf("%d", &n);

	Cấp phát vùng nhớ cho biến dayso	int *dayso = (int *)malloc(n * sizeof(int));
	Gọi hàm tạo dãy số ngẫu nhiên	taodaysongaunhien(dayso, n);
	In dãy số vừa tạo để kiểm tra thông tin nhập	for (int i = 0; i < n; i++) { printf("%3d ", dayso[i]);
	Khai báo khai báo và khởi tạo một con trỏ (pointer) trỏ tới một chuỗi ký tự không thay đổi (constant string) có giá trị là "dayso.dat".	const char *filename = "dayso.dat";
	Gọi hàm ghi dãy số vào tệp	ghidaysovaotep(dayso, n, filename);
	Gọi hàm đọc dãy số từ tệp	printf("Doc va in noi dung tu file:\n"); docdaysotutep(dayso, &n, filename);
	Gọi hàm in dãy số để mình dễ kiểm tra	indayso(dayso, n);

	Gọi hàm sắp xếp dãy số tăng	printf("Sap xep tang dan: ");
		sapxepdaysotang(dayso, n);
		indayso(dayso, n);

	Gọi hàm In kết quả sau khi sắp xếp
	
	Gọi hàm sắp xếp dãy số giảm dần
	Gọi hàm In kết quả để kiểm tra
	
	Giải phỏng bộ nhớ
		printf("Sap xep giam dan: ");
		sapxepdaysogiam(dayso, n);
		indayso(dayso, n);
		
		free(dayso);
		
		getch();
		}

Bài 5. Để quản lý dữ liệu về học sinh, ta cần các thông tin sau: mã học sinh, họ tên, năm sinh, quê quán. Yêu cầu:

- Khai báo dữ liệu để lưu trữ đúng và đầy đủ những thông tin trên.
- Viết hàm nhập danh sách học sinh vào mảng.
- Viết hàm ghi danh sách học sinh từ mảng vào tập tin có tên là DATA.INP
- Viết hàm hiện tất cả thông tin của các học sinh từ mảng lên màn hình.
- Viết hàm đọc danh sách học sinh từ tập tin DATA.INP ra mảng.
- Viết hàm tìm một học sinh có mã học sinh cho trước
- Viết hàm chèn một học sinh vào danh sách.
- Viết hàm xóa một học có mã học cho trước ra khỏi danh sách.
- Viết hàm sắp xếp các danh sách học sinh giảm dần dựa theo mã học sinh.

Bài tập số 5 các em làm tương tự với bài tập mẫu đã có code kết hợp với các bài tập bên bài cấu trúc cô đã hướng dẫn