**프로그래밍 기초 과제**

**12주차 실습**

|  |  |
| --- | --- |
| 학과 | 컴퓨터공학과 |
| 학번 | 2022111120 |
| 이름 | 김지민 |
| 담당교수님 | 한인 교수님 |
| 제출일자 | 2023.05.29.월 |

**| 배운 내용 요약**

**- 배열과 함수**

: 배열의 경우에는 사본이 아닌 원본(주소)이 전달된다. (포인터로 보내면 원본이 바뀐다.)

→ 값을 변경하면 원래 배열도 바뀐다.

: 배열은 복사할 수 없다.

🡪 배열에 값이 너무 많은데, 기본적으로 몇 개인지 모르기 때문에 복사할 수 없다.

🡪 따라서 배열은 포인터로 받는다.

: 배열의 이름 = 배열의 시작 위치 주소이다.

🡪 시작 위치 주소만 보내기 때문에, 배열의 끝이 어디인지 알 수 없다.

🡪 따라서 배열을 보낼 때, int n과 같이 배열의 사이즈도 보내줘야 한다. (끝을 찾아가기 위함)

: 배열을 포인터로 보내는 방법

1) int a[ ] 배열 형식 유지 🡨 이 방법을 추천한다.

2) int \* a

**| 실습 문제 1**

| 코드 설명 :

- 배열에 저장할 자료형의 타입이 double이므로, double로 선언했습니다.

- 벡터X와 벡터Y를 각각의 일차원 배열로 설정한 후, scanf를 통해 값을 입력 받습니다.

(1) dot\_prod\_result 함수 선언

: 벡터 계산을 진행하는 함수를 선언합니다.

: 이 함수는 배열을 전달받습니다. 그래서 매개변수를 double x[], double y[]로 설정했습니다.

: 배열의 끝을 알기 위한 사이즈를 따로 보내지 않고, 상수로 선언한 SIZE를 활용하여 배열의 값에 접근했습니다.

: 벡터 내적 결과값을 return 합니다. 이 값을 메인 함수에서 dot\_prod\_result에 저장합니다.

(2) print\_func 함수 선언

: 결과값을 프린트하는 함수를 선언합니다.

: X,Y 값을 갖고 있는 일차원 배열을 받고, dot\_prod\_result함수로 나온 결과를 저장한 dot\_prod\_result의 값을 double형으로 받습니다.

**| 출력 결과**

|  |
| --- |
|  |

**| 실습문제2**

**| 코드 설명 :**

**1) 랜덤으로 좌표 생성하기**

좌표를 저장할 coordinate이름의 이차원 배열을 선언합니다. 처음에는 0으로 초기화를 했습니다.

-100부터 100까지 사이의 난수를 생성하기 위해 rand() % (200+1)-100을 해줬습니다.

난수 함수를 통해 만들어낸 숫자를 우선 Tmp\_x와 Tmp\_y에 각각 저장을 합니다. 바로 배열에 넣지 않는 이유는 중복되는 숫자가 있는지를 확인하기 위함입니다.

**2) 삽입정렬 기능의 함수 생성하기**

inserSort 함수는 이차원 배열을 매개변수로 전달받습니다.

[1] targetIdx

삽입정렬은 2번째 원소부터 비교를 시작하기 때문에, targetIdx= i+1로 수식을 설정해줍니다.

targetIdx는 비교를 할 key원소의 위치에 접근하는 용도입니다.

[2] 첫 번째 반복문의 조건식 ROW-1

삽입정렬은 두번째 원소부터 비교를 시작해나가기 때문에 n개의 원소가 있으면, n-1번 시행되어야 해서 ROW-1을 조건식으로 설정했습니다.

[3] 두 번째 반복문

삽입정렬은 key 원소보다 하나 아래에 있는 위치부터 첫번째 원소까지 거꾸로 비교를 해야합니다. 그래서 비교대상에 접근하기 위한 두 번째 반복문에서는 j를 targetIdx로 초기식을 설정하고, j작아지도록 증감식을 설정했습니다.

[3-1] 첫 번째 if문

비교의 기준이 되는 원소인 targetIdx와, 그 보다 하나 앞에 있는 수와 비교를 하기 위해서 coordinate[j][0]과 coordinate[j-1][0]을 비교해줍니다. coordinate[j][0]이 더 작다면, j번째 좌표와 j-1번째 좌표를 바꿔줍니다. X와 y모두 바꿔줍니다.

[3-2] else-if 문

coordinate[j][0]과 coordinate[j-1][0] 가 같다면, X좌표가 같다는 의미이므로, Y좌표를 비교해줍니다. 동일하게 [3-1]과 동일하게 Y좌표를 기준으로 오름차순 정렬을 해줍니다.

[3-3] else 문

여기에는 coordinate[j][0]이 coordinate[j-1][0]보다 큰 경우가 걸리게 됩니다. 이 때는 현재의 정렬을 유지하면 되므로 continue로 처리했습니다.

**| 출력 결과**

|  |
| --- |
|  |