

자료구조

3 주차 과제



제출일자 : 2023 년 03 월 22 일

담당교수 : 엄태훈 교수님

학번 : 2020127027

이름 : 변영준

[문제 1] 행렬 A와 행렬 B를 입력받아 덧셈과 뺄셈 연산의 결과를 출력하세요. (10 점)

1. 문제 제기

(1) 행렬 A와 행렬 B를 입력받아 각각의 행렬을 입력받은 원소로 채우는 프로그램 구현

-행의 수와 열의 수는 1~10의 범위에 속하도록 프로그램 작성

-행렬의 원소들은 -100~100의 원소들만 입력받도록 프로그램 작성

(2) 행렬 A와 행렬 B의 행과 열의 수가 같을 때만 연산이 가능하도록 하는 프로그램 구현

-만약 두 행렬의 행과 열의 수가 다를 경우 강제 종료하는 프로그램 작성

(3) 행렬 A와 행렬 B의 덧셈과 뺄셈 연산을 수행한 후 출력하는 프로그램 구현

-덧셈과 뺄셈 결과를 순서대로 출력하는 프로그램 작성

2. 문제 해결 과정

(1) 행렬 A와 행렬 B를 입력받아 각각의 행렬을 입력받은 원소로 채우는 프로그램 구현

-행의 수와 열의 수와 행렬의 원소들을 주어진 범위내에서만 입력받을 수 있도록 하는 코드 작성

(2) 행렬 A와 행렬 B의 행과 열의 수가 같을 때만 연산이 가능하도록 하는 프로그램 구현

-R1과 R2를 비교하고 C1과 C2를 비교함으로써 행렬의 연산조건을 맞춘다

(3) 행렬 A와 행렬 B의 덧셈과 뺄셈 연산을 수행한 후 출력하는 프로그램 구현

-for 문을 이중 중첩한 것을 연달아 2개 사용함으로써 덧셈과 뺄셈의 결과를 순서대로 출력

3. 결과

소스코드	<pre>#include <iostream> #include <vector> using namespace std; int A[10001][10001] = {0,}; //충분히 큰 크기의 2차원 배열 설정 int B[10001][10001] = {0,}; //충분히 큰 크기의 2차원 배열 설정 int main() { /*배열 생성하는 코드*/ int R1, C1, R2, C2; /*배열 A 생성 코드*/ cout << "만들고자 하는 행렬 A의 행 크기와 열 크기를 공백을 기준으로 입력하세요" << endl; cin >> R1 >> C1; if (R1 > 10 R1 < 1 C1 > 10 C1 < 1) { cout << "반드시 행렬 A의 행 크기와 열 크기는 각각 1~10이어야 합니다." << endl; return -1; } cout << "행렬 A에 넣을 원소들을 공백을 기준으로 입력하세요(입력하신 행의 크기와 열의 크기를 고려하여 데이터를 입력하시오)" << endl; cout << "가령, 3x3 행렬의 경우 아래와 같이 입력하시오. Wn2 3 4Wn2 3 4Wn2 3 4" << endl; for(int i = 0; i < R1; i++) for (int j = 0; j < C1; j++) { cin >> A[i][j]; if (A[i][j] < -100 A[i][j] > 100) { cout << "반드시 행렬의 원소는 -100~100 사이여야 합니다." << endl; return -1; } } //배열 A 생성 완료 /*배열 B 생성 코드*/ cout << "만들고자 하는 행렬 B의 행 크기와 열 크기를 공백을 기준으로 입력하세요" << endl; cin >> R2 >> C2; if (R2 > 10 R2 < 1 C2 > 10 C2 < 1) { cout << "반드시 행렬 B의 행 크기와 열 크기는 각각 1~10이어야 합니다." << endl; return -1; } cout << "행렬 B에 넣을 원소들을 공백을 기준으로 입력하세요(입력하신 행의 크기와 열의 크기를 고려하여 데이터를 입력하시오)" << endl; cout << "가령, 3x3 행렬의 경우 아래와 같이 입력하시오. Wn2 3 4Wn2 3 4Wn2 3 4" << endl; for (int i = 0; i < R1; i++) for (int j = 0; j < C1; j++) { cin >> B[i][j]; if (B[i][j] < -100 B[i][j] > 100) { cout << "반드시 행렬의 원소는 -100~100 사이여야 합니다." << endl; } } << endl;</pre>
------	---

```

        return -1;
    }
}
cout << "WnWnWn" << endl;
//배열 B 생성 완료

if (R1 != R2 | C1 != C2) {
    cout << "계산 오류" << endl;
    return -1;
} //A와 B의 행의 수와 열의 수가 각각 같을 때만 연산이 되도록 설정
else {
    for (int i = 0; i < R1; i++) {
        for (int j = 0; j < C1; j++) {
            int temp = A[i][j] + B[i][j];
            cout << temp << " ";
        }
        cout << " " << endl;
    } //A+B의 행렬 연산
    cout << "WnWnWnWnWn" << endl;
    for (int i = 0; i < R1; i++) {
        for (int j = 0; j < C1; j++) {
            int temp = A[i][j] - B[i][j];
            cout << temp << " ";
        }
        cout << " " << endl;
    } //A-B의 행렬 연산
}

return 0;
}

```

<p>실행화면</p>	<div data-bbox="459 230 1316 1122"> <p>만들고자 하는 행렬 A의 행 크기와 열 크기를 공백을 기준으로 입력하세요</p> <p>3 3</p> <p>행렬 A에 넣을 원소들을 공백을 기준으로 입력하세요(입력하신 행의 크기와 열의 크기를 고려하여 데이터를 입력하시오)</p> <p>가령, 3x3 행렬의 경우 아래와 같이 입력하시오.</p> <p>2 3 4</p> <p>2 3 4</p> <p>2 3 4</p> <p>7 7 7</p> <p>7 7 7</p> <p>7 7 7</p> <p>만들고자 하는 행렬 B의 행 크기와 열 크기를 공백을 기준으로 입력하세요</p> <p>3 3</p> <p>행렬 B에 넣을 원소들을 공백을 기준으로 입력하세요(입력하신 행의 크기와 열의 크기를 고려하여 데이터를 입력하시오)</p> <p>가령, 3x3 행렬의 경우 아래와 같이 입력하시오.</p> <p>2 3 4</p> <p>2 3 4</p> <p>2 3 4</p> <p>2 2 2</p> <p>3 3 3</p> <p>4 4 4</p> <p>9 9 9</p> <p>10 10 10</p> <p>11 11 11</p> <p>5 5 5</p> <p>4 4 4</p> <p>3 3 3</p> <p>Windows 7</p> </div> <div data-bbox="459 1155 1382 1839"> <p>만들고자 하는 행렬 A의 행 크기와 열 크기를 공백을 기준으로 입력하세요</p> <p>3 3</p> <p>행렬 A에 넣을 원소들을 공백을 기준으로 입력하세요(입력하신 행의 크기와 열의 크기를 고려하여 데이터를 입력하시오)</p> <p>가령, 3x3 행렬의 경우 아래와 같이 입력하시오.</p> <p>2 3 4</p> <p>2 3 4</p> <p>2 3 4</p> <p>7 7 7</p> <p>8 8 8</p> <p>9 9 9</p> <p>만들고자 하는 행렬 B의 행 크기와 열 크기를 공백을 기준으로 입력하세요</p> <p>4 4</p> <p>행렬 B에 넣을 원소들을 공백을 기준으로 입력하세요(입력하신 행의 크기와 열의 크기를 고려하여 데이터를 입력하시오)</p> <p>가령, 3x3 행렬의 경우 아래와 같이 입력하시오.</p> <p>2 3 4</p> <p>2 3 4</p> <p>2 3 4</p> <p>1 2 3 4</p> <p>5 6 7 8</p> <p>9 10 11 12</p> <p>계산 오류</p> </div>
-------------	---

4. 결론

이번 과제는 배열의 연산 특히나 for 구문을 사용하여 직접 배열의 원소들을 가지고 코딩을 할 수 있는 것에 대해 연습하는 과제였다. 아직은 C++언어를 사용하여 예외처리를 하는 부분에 대해선 수업에서 배운 적이 없어 구현하진 못했지만 만일 예외처리를 사용하는 코드를 짤 수 있다면 보다 더 객체 지향 프로그래밍을 더 잘 구현할 수 있을 것이다. 이 부분은 앞으로 수업을 들어가며 보충할 예정이다.

[문제 2] 정수 N 개로 주어진 수열 A 와 정수 X가 주어질때,
A 에서 X 보다 큰 수를 모두 출력하는 프로그램을 작성하세요.
(10 점)

1. 문제 제기

(1) 배열을 생성하고 배열의 원소를 입력받는 프로그램 구현

-첫 줄에 공백을 기준으로 A 배열의 크기 N 과 X 를 입력받도록 프로그램 작성

-둘째 줄에 공백을 기준으로 수열의 첫번째 원소부터 입력받도록 프로그램 작성

-입력받는 모든 값은 10000 보다 작거나 같은 자연수만 입력받도록 프로그램 작성

(2) 입력된 배열의 원소들이 오름차순 정렬이 되도록 프로그램 구현

(3) 오름차순으로 정렬이 된 프로그램에서 입력된 값(x)보다 큰 값을 찾는 프로그램 구현

-배열에서 x 보다 큰 수가 없을 경우 Error 를 출력하고 프로그램을 강제 종료하도록 프로그램 작성

2. 문제 해결 과정

(1) 배열을 생성하고 배열의 원소를 입력받는 프로그램 구현

-배열의 크기와 배열의 원소는 반드시 10000 보다 작거나 같은 자연수가
입력되도록 예외처리를 진행

(2) 입력된 배열의 원소들이 오름차순 정렬이 되도록 프로그램 구현

-선택 정렬 알고리즘을 사용

(3) 오름차순으로 정렬이 된 프로그램에서 입력된 값(x)보다 큰 값을 찾는
프로그램 구현

-이진 탐색 알고리즘을 사용


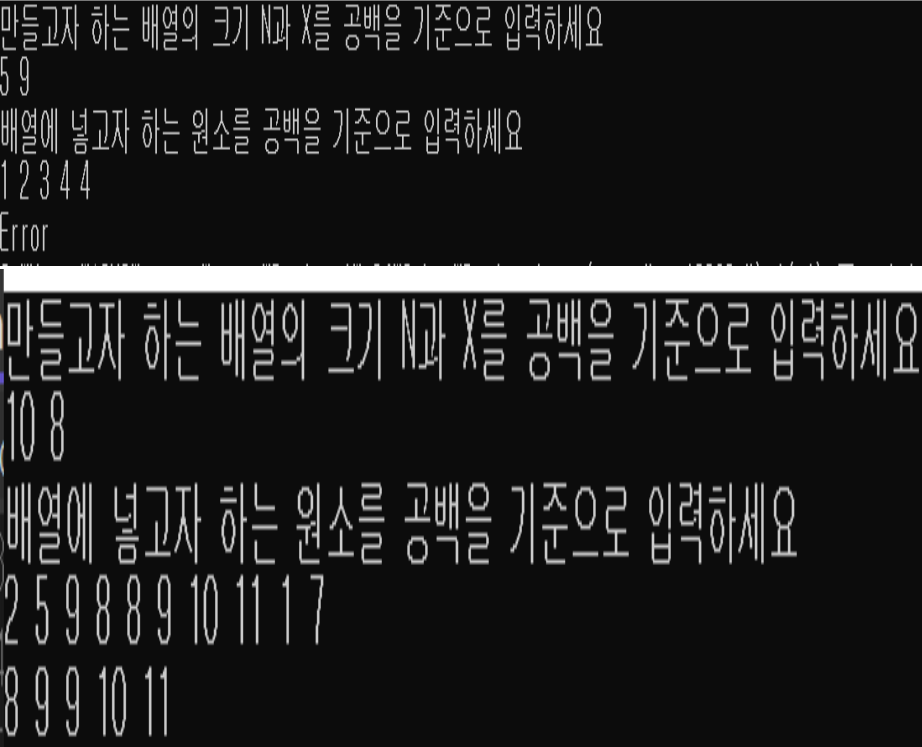
-이진 탐색 알고리즘을 사용한 후 결과값으로 도출된 middle 번째의 원소가 이
배열에 포함되어 있지 않아도 오류가 없이 큰 값을 찾도록 함

-주어진 배열에서 더 큰 값이 없다면 에러를 출력한 후 강제 종료함

-항상 찾은 값보다 큰 값만 출력하므로 middle 에 1 을 더한 값이 배열의
크기(N)보다 작은지 비교하는 과정 진행

3. 결과

소스코드	<pre>#include <iostream> #include <vector> using namespace std; int A[10001] = { 0, }; //주어진 배열의 최대로 설정 int main() { /*배열 생성하는 코드*/ int N, X; cout << "만들고자 하는 배열의 크기 N과 X를 공백을 기준으로 입력하세요" << endl; cin >> N >> X; if (N > 10000 X > 10000 N < 0 X < 0) { cout << "N과 X는 반드시 10000보다 작거나 같은 자연수이어야 합니다!"<<endl; return -1; } cout << "배열에 넣고자 하는 원소를 공백을 기준으로 입력하세요" << endl; for (int i = 0; i < N; i++) { cin >> A[i]; if (A[i] > 10000 A[i] < 0) { cout << "A의 원소는 반드시 10000보다 작거나 같은 자연수이어야 합니다!"<<endl; return -1; } } //배열 생성 완료 /*선택 정렬하는 코드*/ for (int i = 0; i < N; i++) { int j = i; for (int k = i + 1; k < N; k++) if (A[k] < A[j]) j = k; swap(A[i], A[j]); } //선택 정렬 완료 /*이진탐색하는 코드*/ int left = 0; int right = N - 1; int middle = (left + right) / 2; while (left <= right) { middle = (left + right) / 2; if (X < A[middle]) right = middle - 1; else if (X > A[middle]) left = middle + 1; else break; } }</pre>
------	--

	<pre> //이진탐색 완료 /*배열 A에서 X보다 큰 수를 출력하는 코드*/ if (middle + 1 < N) { //찾은 값보다 큰 값만 출력하므로 배열에서 +1하고 그 값이 배열의 크기보다 작아야함 for (int i = middle + 1; i < N; i++) cout << A[i] << " "; } else { //주어진 배열에서 더 큰 값이 없다면 에러를 출력한 후 강제 종료 cout << "Error"; return -1; } //출력 완료 return 0; } </pre>
<p>실행화 면</p>	<p> Microsoft Visual Studio 디버그 콘솔</p>  <p>만들고자 하는 배열의 크기 N과 X를 공백을 기준으로 입력하세요 5 9 배열에 넣고자 하는 원소를 공백을 기준으로 입력하세요 1 2 3 4 4 Error</p> <p>만들고자 하는 배열의 크기 N과 X를 공백을 기준으로 입력하세요 10 8 배열에 넣고자 하는 원소를 공백을 기준으로 입력하세요 2 5 9 8 8 9 10 11 1 7 8 9 9 10 11</p>

4. 결론

이번 과제는 수업시간에 학습 했었던 선택 정렬 알고리즘과 이진 탐색 알고리즘과 더불어 C++의 문법들을 복습하고, 프로그래밍 연습을 할 수 있는 과제였다. 평소 사용하던 C 언어와 문법이 비슷하면서도 약간은 다른 C++ 특유의 문법에 더 익숙해지게 되는 계기가 되었으며 이번 과제를 진행하게 되며 실제 손으로 그려가며 알고리즘에 따라서 데이터가 어떠한 flow 를 가지고 접근되어지는가를 직접 눈으로 보가며 그 흐름을 따라갔고 그 결과 이진 탐색 알고리즘의 경우 만약 내가 찾고자 하는 값이 배열의 원소보다 크다면 항상 middle 은 N-2 의 원소 즉 제일 마지막의 앞에 있는 원소에서 언제나 정지되어 있는 것을 확인 하였고, 그와는 별개로 만약 배열의 원소들이 모두 다 내가 찾고하는 값보다 크다면 언제나 middle 은 0 번째에서 멈추는 것을 확인 하였다. 알고리즘 자체에 내장되어 있는 문제인 것 같아 이 부분은 앞으로 알고리즘 최적화를 배워가며 더 보완할 예정이다.