Trạng thái	Đã xong
Bắt đầu vào lúc	Thứ Tư, 16 tháng 4 2025, 5:06 PM
Kết thúc lúc	Thứ Tư, 16 tháng 4 2025, 5:17 PM
Thời gian thực hiện	10 phút 50 giây

Câu hỏi 1

Đúng

Mô tả tiếng Việt:

Hãy hiện thực hàm readArray() được khai báo như sau:

int** readArray()

Hàm này sẽ đọc dữ liệu cho một ma trận 2 chiều, mỗi chiều có 10 phần tử. Các phần tử của ma trận sẽ được nhập vào từ bàn phím (từ phần tử a[0][0] cho đến a[9][9]). Tuy nhiên nếu phần tử a[i][j] được nhập là 0 thì tất cả các phần tử còn lại trên hàng (a[i][k], j<k<10) đều được tự động gán là là 0, chương trình sẽ đọc tiếp phần tử a[i+1][0] từ bàn phím. Hàm readArray sẽ trả về một con trỏ tới mảng 2 chiều đã nhập này.

Đầu vào: Các phần tử có trong mảng 2 chiều, mỗi phần tử là một số nguyên dương có giá trị không vượt quá 1000.

Đầu ra: Con trỏ tới mảng 2 chiều vừa tạo

English version:

Implement the function readArray() that is declared as below syntax:

int** readArray()

The function reads a two-dimensional matrix each of which consists of 10 elements. These elements are entered from the keyboard (from a[0][0] to a[9][9]). If a[i][j] is assigned to 0, all remained element of the row (a[i][k], j<k<10) will automatically assigned to 0, and the function will continue to input the next-row element from the keyboard. Moreover, this function also returns a pointer which points to the two-dimensional matrix just entered.

Input: The positive integer matrix's elements which not surpass 1000.

Output: The pointer that points to the two-dimensional matrix just entered.

For example:

Test	Input	Result
1	1 2 3 4 5 6 7 8 9 10 0 1 0 2 0 3 0 4 5 0 6 7 0 8 0 9 0 10 11 12 13 14 0	1 2 3 4 5 6 7 8 9 10 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 4 5 0 0 0 0 0 0 0 0 6 7 0 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 0 9 0 0 0 0 0 0 0 0 0 10 11 12 13 14 0 0 0 0 0
2	0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Answer: (penalty regime: 0 %)

Reset answer

```
int** readArray() {
    // Cấp phát bộ nhớ cho ma trận 10x10
    int** matrix = new int*[10];
    for (int i = 0; i < 10; i++) {
        matrix[i] = new int[10];
    }
```

```
    }

    // Đọc dữ liệu ma trận
    for (int i = 0; i < 10; i++) {
        for (int j = 0; j < 10; j++) {
            // Đọc phần tử a[i][j]
            cin >> matrix[i][j];

            // Nếu phần tử là 0, tự động gán tất cả phần tử còn lại trong hàng bằng 0
            if (matrix[i][j] == 0) {
                // Gán các phần tử còn lại trong hàng là 0
                for (int k = j + 1; k < 10; k++) {
                    matrix[i][k] = 0;
                }
                // Thoát khỏi vòng lặp j và chuyển sang hàng tiếp theo
                break;
            }
        }
    }

    // Trả về con trỏ đến ma trận
    return matrix;
}
```

	Test	Input	Expected	Got	
✓	1	1 2 3 4 5 6 7 8 9 10 0 1 0 2 0 3 0 4 5 0 6 7 0 8 0 9 0 10 11 12 13 14 0	1 2 3 4 5 6 7 8 9 10 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 4 5 0 0 0 0 0 0 0 0 6 7 0 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 0 9 0 0 0 0 0 0 0 0 0 10 11 12 13 14 0 0 0 0 0	1 2 3 4 5 6 7 8 9 10 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 4 5 0 0 0 0 0 0 0 0 6 7 0 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 0 9 0 0 0 0 0 0 0 0 0 10 11 12 13 14 0 0 0 0 0	✓

	Test	Input	Expected	Got	
✓	2	0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	✓

Passed all tests! ✓

Câu hỏi **2**

Đúng

Mô tả tiếng Việt:

Hiện thực hàm **void addElement(int*& arr, int n, int val, int index)** nhận vào một mảng động arr có chính xác n phần tử và tiến hành chèn giá trị val vào vị trí thứ index.

Đầu vào: Mảng một chiều arr có kích thước n, giá trị cần chèn val và vị trí cần chèn index.

Đầu ra: Mảng arr sau khi chèn.

Lưu ý: Việc chèn phần tử vào mảng động phải được thực hiện bằng cách giải phóng mảng cũ có n phần tử và cấp phát mảng mới có n+1 phần tử.

English version:

Implement the function **void addElement(int*& arr, int n, int val, int index)** that inputs a dynamic array, arr, consisting of exactly n elements and insert a value, val, into the a specific position, index.

Input: The n-size dynamic array needs to be inserted the value, val, into the specific position, index.

Output: The dynamic array after insert.

Note: Insertion of elements into a dynamic array must be executed by freeing the old array and allocating new memory for the new one.

For example:

Test	Input	Result
1	2 2 3 1 1	2 1 3
2	2 2 3 1 2	2 3 1

Answer: (penalty regime: 0 %)

Reset answer

```
void addElement(int*& arr, int n, int val, int index) {
    // Cấp phát mảng mới có n+1 phần tử
    int* newArr = new int[n + 1];

    // Sao chép các phần tử từ vị trí 0 đến index-1 vào mảng mới
    for (int i = 0; i < index; i++) {
        newArr[i] = arr[i];
    }

    // Chèn giá trị val vào vị trí index
    newArr[index] = val;

    // Sao chép các phần tử còn lại từ vị trí index đến n-1 của mảng cũ
    // vào vị trí index+1 đến n của mảng mới
    for (int i = index; i < n; i++) {
        newArr[i + 1] = arr[i];
    }

    // Giải phóng bộ nhớ của mảng cũ
    delete[] arr;

    // Cập nhật con trỏ arr để trỏ đến mảng mới
    arr = newArr;
}
```

	Test	Input	Expected	Got	
✓	1	2 2 3 1 1	2 1 3	2 1 3	✓

Passed all tests! ✓

▸ Show/hide question author's solution (Cpp)

Câu hỏi **3**

Đúng

Mô tả tiếng Việt:

Hiện thực hàm **int* flatten(int** matrix, int r, int c)** trả về một mảng một chiều được "làm phẳng" từ mảng hai chiều có kích thước r x c (bằng cách nối các hàng của mảng hai chiều lại với nhau).

Đầu vào: Mảng hai chiều có kích thước r x c.

Đầu ra: Mảng một chiều sau khi được "làm phẳng" từ mảng hai chiều đầu vào.

English version:

Implement the function **int* flatten(int** matrix, int r, int c)** tht returns a one-dimensional array flatten from a two-dimensional matrix of size r x c (by concating all the matrix rows).

Input: The two-dimensional matrix of size r x c

Output: The one-dimensional array flatten from the previous two-dimensional matrix.

For example:

Test	Input	Result
1	2 3 1 2 3 4 5 6	1 2 3 4 5 6
2	2 3 1 2 3 4 0 0	1 2 3 4 0 0
3	3 3 1 2 3 4 5 6 2 9 -99	1 2 3 4 5 6 2 9 -99
4	3 4 1 2 3 4 4 5 6 0 -1 8 8 100	1 2 3 4 4 5 6 0 -1 8 8 100
5	4 4 1 2 4 4 4 5 3 0 2 5 1 6 7 7 8 4	1 2 4 4 4 5 3 0 2 5 1 6 7 7 8 4
1	4 1 1 4 2 3	1 4 2 3
7	1 4 1 2 4 4	1 2 4 4

Answer: (penalty regime: 0 %)

Reset answer

```
int* flatten(int** matrix, int r, int c) {
    // Cấp phát mảng một chiều mới có kích thước r*c
    int* result = new int[r * c];

    // Biến đếm để theo dõi vị trí trong mảng result
    int index = 0;

    // Duyệt qua từng phần tử của ma trận hai chiều
    for (int i = 0; i < r; i++) {
        for (int j = 0; j < c; j++) {
            // Sao chép phần tử từ ma trận vào mảng result
            result[index] = matrix[i][j];
            index++;
        }
    }
```

```
16
17      // Trả về con trỏ đến mảng đã được làm phẳng
18      return result;
19  }
```

	Test	Input	Expected	Got	
✓	1	2 3 1 2 3 4 5 6	1 2 3 4 5 6	1 2 3 4 5 6	✓

Passed all tests! ✓

Câu hỏi **4**

Đúng

Mô tả tiếng Việt:

Hiện thực hàm **char* concatStr(char* str1, char* str2)** trả về một chuỗi là kết quả sau khi nối 2 chuỗi str1 và str2 thành một chuỗi duy duy nhất.

Đầu vào: Hai chuỗi str1 và str2.

Đầu ra: Chuỗi được nỗi từ 2 chuỗi con str1 và str2.

Lưu ý: Không được phép sử dụng các hàm hỗ trợ của thư viện string và string.h cho bài tập này.

English version:

Implement the function **char* concatStr(char* str1, char* str2)** that return a string merged from two smaller string str1 and str2.

Input: Two string str1 and str2.

Output: The string merged from two smaller string str1 and str2.

Note: The string and string.h library are not allowed to use for this exercise.

For example:

Test	Result

`char s1[] = "Hello, ";`
`char s2[] = "how are you?";`
`char* s = concatStr(s1, s2);`
`cout << s;`
`delete[] s;`
	`Hello, how are you?`

`char s1[] = "Nice to ";`
`char s2[] = "meet you.";`
`char* s = concatStr(s1, s2);`
`cout << s;`
`delete[] s;`
	`Nice to meet you.`

`char s1[] = "Nice ";`
`char s2[] = "to meet ";`
`char s3[] = "you.";`
`char* temp = concatStr(s1, s2);`
`char* s = concatStr(temp, s3);`
`cout << s;`
`delete[] s; delete[] temp;`
	`Nice to meet you.`

`char s1[] = "Ho Chi Minh ";`
`char s2[] = "University ";`
`char s3[] = "of Technology.";`
`char* temp = concatStr(s1, s2);`
`char* s = concatStr(temp, s3);`
`cout << s;`
`delete[] s; delete[] temp;`
	`Ho Chi Minh University of Technology.`

`char s1[] = "This question ";`
`char s2[] = "is as easy as ";`
`char s3[] = "the other.";`
`char* temp = concatStr(s1, s2);`
`char* s = concatStr(temp, s3);`
`cout << s;`
`delete[] s; delete[] temp;`
	`This question is as easy as the other.`

`char s1[] = "That's ";`
`char s2[] = "a good idea.";`
`char* s = concatStr(s1, s2);`
`cout << s;`
`delete[] s;`
	`That's a good idea.`

`char s1[] = "123";`
`char s2[] = "456";`
`char* s = concatStr(s1, s2);`
`cout << s;`
`delete[] s;`
	`123456`

Test	Result
char s1[] = ""; char s2[] = "CSE"; char* s = concatStr(s1, s2); cout << s; delete[] s;	CSE
char s1[] = ""; char s2[] = ""; char* s = concatStr(s1, s2); cout << s; delete[] s;	

Answer: (penalty regime: 0 %)

Reset answer

```
char* concatStr(char* str1, char* str2) {
    // Tính độ dài của hai chuỗi
    int len1 = 0;
    int len2 = 0;

    // Tính độ dài của chuỗi str1
    while (str1[len1] != '\0') {
        len1++;
    }

    // Tính độ dài của chuỗi str2
    while (str2[len2] != '\0') {
        len2++;
    }

    // Cấp phát bộ nhớ cho chuỗi kết quả (len1 + len2 + 1 cho ký tự null)
    char* result = new char[len1 + len2 + 1];

    // Sao chép chuỗi str1 vào result
    for (int i = 0; i < len1; i++) {
        result[i] = str1[i];
    }

    // Sao chép chuỗi str2 vào sau str1 trong result
    for (int i = 0; i < len2; i++) {
        result[len1 + i] = str2[i];
    }

    // Thêm ký tự kết thúc chuỗi
    result[len1 + len2] = '\0';

    return result;
}
```

Passed all tests! ✓

Câu hỏi **5**

Đúng

Mô tả tiếng Việt:

Chuyển vị của một ma trận 2 chiều là một phần quan trọng trong việc tính toán trên ma trận nói riêng và đại số tuyến tính nói chung.

Gọi B là ma trận sau khi chuyển vị của ma trận A thì ma trận B có tính chất là b[i][j] = a[j][i].

Hãy viết hàm **int** transposeMatrix(int** matrix, int r, int c)** thực hiện phép chuyển vị trên ma trận đã được đề cập bên trên.

Đầu vào:

- Con trỏ tới mảng 2 chiều. Mỗi phần tử trong mảng 2 chiều có giá trị trong khoảng (-1000; 1000).
- Kích thước mảng 2 chiều là 1 cặp số dương r, c. Trong đó: r là số hàng của ma trận, c là số cột của ma trận. Giá trị n không vượt quá 1000.

Đầu ra: Con trỏ trỏ tới mảng hai chiều sau khi được chuyển vị. trong trường hợp ma trận đầu vào rỗng, trả về con trỏ null.

English version:
Transposition of a two-dimensional matrix is an important term for matrix calculations in particular and linear algebra in general.
A matrix B transposed from a matrix A that satisfied the following formula b[i][j] = a[j][i].
Implement the function **int** transposeMatrix(int** matrix, int r, int c)** that perform the transposition of the matrix mentioned above.
Input:

- The pointer that points to a two-dimensional matrix each of whose elements is in the range (-1000; 1000).
- The size of the matrix consists of the number of row r and the number of column n.

Output: The pointer that points to transposed two-dimensional matrix. If the input matrix is empty, return the null pointer.

For example:

Test	Input	Result
1	2 2 1 2 3 4	1 3 2 4
2	1 1 1	1
3	3 3 1 2 3 4 5 6 7 8 9	1 4 7 2 5 8 3 6 9
4	4 4 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	1 5 9 13 2 6 10 14 3 7 11 15 4 8 12 16
5	2 2 10 12 14 16	10 14 12 16
6	2 3 1 2 3 4 5 6	1 4 2 5 3 6
7	1 3 1 2 3	1 2 3
8	3 1 1 1 2	1 1 2
9	0 0	NULL

Test	Input	Result
10	1 2 1 2	1 2

Answer: (penalty regime: 0 %)

Reset answer

```
// Phép chuyển vị của ma trận 2 chiều.
int** transposeMatrix(int** matrix, int r, int c) {
    // Kiểm tra ma trận rỗng
    if (r == 0 || c == 0) {
        return NULL;
    }

    // Cấp phát bộ nhớ cho ma trận kết quả kích thước c x r
    int** result = new int*[c];
    for (int i = 0; i < c; i++) {
        result[i] = new int[r];
    }

    // Thực hiện phép chuyển vị: b[i][j] = a[j][i]
    for (int i = 0; i < c; i++) {
        for (int j = 0; j < r; j++) {
            result[i][j] = matrix[j][i];
        }
    }

    return result;
}
```

	Test	Input	Expected	Got	
✓	1	2 2 1 2 3 4	1 3 2 4	1 3 2 4	✓

Passed all tests! ✓