Project 0 report

선형대수학 0031-04 건국대학교 202210748 문진혁

1. **Write the following functions**

   (a) transposeMatrix(A,m,n): transpose the m $\times$ n matrix A and return the result

```
double** transposeMatrix(double **A, int m, int n) {
    double** B = allocateMemory(n, m);

    for (int i = 0; i < m; i++)
        for (int j = 0; j < n; j++)
            B[j][i] = A[i][j];

    return B;
}
```

   (b) normalizeVector(v,n) normalize the n-dimensional vector v and return the result

```
double** normalizeVector(double** v, int m) {
    double** w;
    double len = 0.0;

    for (int i = 0; i < m; i++)
        len += v[i][0]*v[i][0];
    len = sqrt(len);

    w = allocateMemory(m,1);
    for (int i = 0; i < m; i++)
        w[i][0] = v[i][0]/len;

    return w;
}
```

   (c) calculateLength(v,n) : calculate the length of the n-dimensional vector v and return the result

```
double calculateLength(double** v, int m) {
    double length = 0.0;
    for (int i = 0; i < m; i++) {
        length += v[i][0] * v[i][0];
    }
    length = sqrt(length);
    return length;
}
```

   (d) scaleMatrix(A,m,n,c) : scale the m *times* n matrix A with scalar c

```
double** scaleMatrix(double** A, int m, int n, double c) {
    double** S = allocateMemory(m, n);
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
```

```
            S[i][j] = A[i][j]*c;
        }
    }
    return S;

}
```

(e) multiplyTwoMatrices(A,m,n,B,l,k): for m × n matrix A and l × k matrix B, calculate and return AB. Return "null" if multiplication is impossible

```
double** multiplyTwoMatrices(double** A, int m, int n, double** B, int l, int k) {
    if (n != l) {
        return NULL;
    }
    double** S = allocateMemory(m, k);
    double temp;

    for (int i = 0; i < m; i++) {
        for (int j = 0; j < k; j++) {
            temp = 0.0;
            for (int p = 0; p < n; p++) {
                temp += A[i][p] * B[p][j];
            }
            S[i][j] = temp;
        }
    }
    return S;
}
```

(f) addTwoMartrices(A,m,n,B,l,k): for m × n matrix A and l × k matrix B, calculate and return $A + B$. Return "null" if multiplication is impossible

```
double** addTwoMatrices(double** A, int m, int n, double** B, int l, int k) {
    if (m != l || n != k) {
        return NULL;

    }
    double** S = allocateMemory(m, n);
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            S[i][j] = A[i][j] + B[i][j];
        }
    }
    return S;
}
```

2. **Write a computer program in C that performs the following**

   (a) Test the correctness of each of the function you wrote in 1.

1. (a) transposeMatrix

```
A =
 1.000    1.000    1.000
 1.000    2.000    3.000
 1.000    3.000    5.000
 1.000    4.000    7.000
```

```
A_transposed =
 1.000    1.000    1.000    1.000
 1.000    2.000    3.000    4.000
 1.000    3.000    5.000    7.000
```

1. (b) normalizeVector

```
v =
 0.000
 1.000
 2.000
 3.000
```

```
v_normalized =
 0.000
 0.267
 0.535
 0.802
```

1. (c) calculateLength

```
v =
 0.000
 1.000
 2.000
 3.000
```

```
(the length of v) = 3.741657
```

1. (d) scaleMatrix

```
A =
 1.000    1.000    1.000
 1.000    2.000    3.000
 1.000    3.000    5.000
 1.000    4.000    7.000
```

```
A_scaled(5.0) =
 5.000    5.000    5.000
 5.000   10.000   15.000
 5.000   15.000   25.000
 5.000   20.000   35.000
```

1. (e) multiplyTwoMatrices

```
A =
 1.000    1.000    1.000
 1.000    2.000    3.000
 1.000    3.000    5.000
 1.000    4.000    7.000
```

```
C =
 0.000    2.000    4.000    6.000
 1.000    3.000    5.000    7.000
 2.000    4.000    6.000    8.000
```

```
multiply A and C =
 3.000    9.000   15.000   21.000
 8.000   20.000   32.000   44.000
13.000   31.000   49.000   67.000
18.000   42.000   66.000   90.000
```

1. (f) addTwoMatrices

```
A =
 1.000    1.000    1.000
 1.000    2.000    3.000
 1.000    3.000    5.000
 1.000    4.000    7.000
```

```
B =
 0.000    2.000    4.000
 1.000    3.000    5.000
 2.000    4.000    6.000
 3.000    5.000    7.000
```

```
add A and B =
 1.000    3.000    5.000
 2.000    5.000    8.000
 3.000    7.000   11.000
 4.000    9.000   14.000
```

(b) For given n × matrices A and $\bar{H}$, normalized each column of $\bar{H}$(let H be this normalized matrix). Then, calculate $B = H^T AH$, and then, $C = HBH^T$.

$$A = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} \text{ and } \bar{H} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \text{,then } H = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$$

$$\begin{aligned} B &= H^T AH \\ &= \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \\ &= \begin{bmatrix} 4.5 & -1.5 \\ -1.5 & 0.5 \end{bmatrix} \end{aligned}$$

$$\begin{aligned} C &= HBH^T \\ &= \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 4.5 & -1.5 \\ -1.5 & 0.5 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \\ &= \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} \end{aligned}$$

(1) A, H 행렬 생성

```
A = allocateMemory(2, 2);
for (int i = 0; i < 2; i++) {
    for (int j = 0; j < 2; j++) {
        A[i][j] = (i+1)*(j+1);
    }
}

H = allocateMemory(2, 2);
H[0][0] = 1/sqrt(2);
H[0][1] =1/sqrt(2);
H[1][0] = 1 / sqrt(2);
H[1][1] = -1 / sqrt(2);
```

```
A =
 1.000     2.000
 2.000     4.000
```

```
H =
 0.707     0.707
 0.707    -0.707
```

(2) $B = H^T AH$

```
B = multiplyTwoMatrices(
        multiplyTwoMatrices(
            transposeMatrix(H, 2, 2),2, 2, A, 2, 2), 2, 2,H, 2, 2);
```

```
B =
 4.500    -1.500
-1.500     0.500
```

(c) $C = HBH^T$

```
C = multiplyTwoMatrices(
        multiplyTwoMatrices(H, 2, 2, B, 2, 2), 2, 2,
            transposeMatrix(H, 2, 2), 2, 2);
```

```
C =
 1.000    2.000
 2.000    4.000
```