Объект хандлагат программчлал лаборатори 6

М.Наранбат 21B1NUM0316

ОРШИЛ/УДИРТГАЛ

C++ хэлний классын хүрээнд оператор дахин тодорхойлох арга аргачилтай танилцан матриц нэртэй класс тодорхойлон матриц дээр хийгддэг үйлдлүүд болох 2 матрицыг үржүүлэх, нэмэх, хасах, 1 матрицыг тогтмол тоогоор ихэсгэх, хорогдуулах, үржүүлэх мөн хөрвүүлэх зэрэг үйлдлүүдийг оператор дахин тодорхойлох замаар хийж гүйцэтгэсэн болно.

ЗОРИЛГО, ЗОРИЛТ

Уг лабораторийн ажлын хүрээнд матриц нэртэй класс тодорхойлон матриц дээр хийгддэг үйлдлүүдийг оператор дахин тодорхойлох замаар хэрэгжүүлэх зорилготой ажилсан болно. Үүний хүрээнд дараах зорилтуудыг тавин ажиллав.

- + /нэмэх/ оператор тодорхойлох. Ингэхдээ уг операторыг 2 янзаар тодорхойлсон ба нэгт R = R1 + 12.5 буюу матриц дээр тогтмол тоо нэмэх мөн R = R1 + R2 буюу 2 матрицыг нэмэн 3 дахь обьектод оноох
- * /үржих/ хоёр матрицийг үржээд үржвэрийг буцаана
- - /хасах/ хоёр матрицийг хасаад ялгаварыг буцаана
- = /утга оноох/ нэг матрицийн объектыг нөгөөд утга оноох
- ++ /нэгээр нэмэгдүүлэх/ матрицын гишүүн бүрийг 1-ээр нэмэгдүүлэх
- -- /нэгээр хорогдуулах/ матрицын гишүүн бүрийг 1-ээр хорогдуулна
- += /нэмэгдүүлэх/ нэг матрицыг нөгөө матрицаар нэмэгдүүлэх
- -= /хорогдуулах/ нэг матрицыг нөгөө матрицаар хорогдуулна
- *=/дахин нэмэгдүүлэх/ нэг матрицыг нөгөө матрицаар үржүүлэн үр дүнг анхны матрицад хадгална.
- Матрицыг хөрвүүлэх

ОНОЛЫН СУДАЛГАА

3.1 Оператор дахин тодорхойлох

Оператор дахин тодорхойлох гэдэг нь нэгэнт тодорхойлогдчихсон байдаг ихэнх операторуудыг класс, обьектын хэмжээнд тодорхойлохыг хэлнэ. Жишээ нь object++ буюу тухайн нэг обьектын зарим нэг гишүүн элементүүдийг нэгээр нэмэгдүүлэх зэргээр ++ операторыг классын хэмжээнд тодорхойлох боломжтой юм. Мөн өөрөөр 2 object хооронд нь нэмэх + операторыг дахин тодорхойлон obj_1 = obj_2 + obj_3 гэх мэтээр ашиглах боломжтой.

Оператор дахин тодорхойлоход дараах хэд хэдэн дүрмийг баримталдаг.

- Огт байдаггүй оператор дахин тодорхойлохгүй байх
- Тодорхойлох гэж байгаа операторыг шинж чанарыг алдагдуулахгүй байх. Жишээ нь ++ операторыг обьектын гишүүн өгөгдлүүдийг хорогдуулах байдлаар ашиглахгүй байх.
- Оперантуудын ядаж нэг нь обьект байх хэрэгтэй.

Мөн дээр дурдсанчлан ихэнх операторуудыг дахин тодорхойлох боломжтой. Өөрөөр хэлбэл зарим операторуудыг дахин тодорхойлох боломжгүй байдаг.

Дахин тодорхойлох боломжтой операторууд:

Operators that can be overloaded	Examples
Binary Arithmetic	+, -, *, /, %
Unary Arithmetic	+, -, ++, —
Assignment	=, +=,*=, /=,-=, %=
Bitwise	& , | , << , >> , ~ , ^
De-referencing	(->)
Dynamic memory allocation, De-allocation	New, delete
Subscript	[ ]
Function call	()
Logical	&, | |, !
Relational	>, < , = =, <=, >=

Дахин тодорхойлох боломжгүй операторууд:

- sizeOf
- typeId
- Scope resolution (::)
- Дан хандалтын оператор (. (dot), *); (geeksforgeeks, 2023)

Классын хэмжээнд оператор дахин тодорхойлоход operator гэсэн түлхүүр үг ашигладаг. Жишээ нь:

```
Matrix operator+(const Matrix &other) const;
```

4. ХЭРЭГЖҮҮЛЭЛТ

4.1 Матриц нэртэй класс тодорхойлох

Гишүүн өгөгдлүүд нь матрицын мөрийн хэмжээ, баганы хэмжээ, матрицын элементүүдийн утгыг хадгалдаг pointer of pointer төрлийн values нэртэй хувьсагч байна.

```
class Matrix
{
private:
    int row, column;
    float **values;

public:
    Matrix();                                   // baiguulagch function
    Matrix(int row, int column);                // paramatert baiguulagch function
    Matrix(const Matrix &other);                // Huulagch function
    ~Matrix();                                  // Ustgagch function
    void print() const;                         // Matrixig hevleh function
    int getRow() const;                         // Matrixin moriig awah function
    int getColumn() const;                      // Matrixin bagana iig awah function
    float getValue(int i, int j) const;         // Matrixin utgiig awah function
    void setValue(int i, int j, float x);       // Matrixin utgiig oorchloh function
    void setRow(int newRow);                    // Matrixin moriin setter function
    void setColumn(int newColumn);              // Matrixin baganiin setter function
    Matrix operator+(float x) const;            // Matrixig nemeh operator function
    Matrix operator+(const Matrix &other) const; // Matrixuudig nemeh operator function
    Matrix operator-(const Matrix &other) const; // Matrixuudig hasah function
    Matrix operator*(const Matrix &other) const; // Matrixuudig urjuuleh function
    Matrix &operator=(const Matrix &other);     // Matrixig onooh function
    Matrix &operator++();                       // Matrixin utgiig 1 eer nemegduuleh function
    Matrix &operator--();                       // Matrixin utgiig 1 eer horogduulah function
    Matrix &operator+=(const Matrix &other);    // Matrixig oor matrixaar nemegduuleh function
    Matrix &operator-=(const Matrix &other);    // Matrixig oor matrixaar horogudulah function
    Matrix &operator*=(const Matrix &other);    // Matrixig oor matrixar urjuuleh function
    void convert();                             // Matrixig horwuuleh function
};
```

4.2 Байгуулагч болон устгагч функц

Параметргүй байгуулагч функцын хувьд матрицийн мөр болон баганы хэмжээ 1, 1 байх болно.

```
Matrix::Matrix() : row(1), column(1)
{
    values = new float *[row];
    values[0] = new float[column];
    values[0][0] = 0;
}
```

4.3 Getter болон Setter функц

Setter функцын хувьд тухайн нэг матрицын гишүүн бүрийг нэг нэгээр өөрчлөх боломжтойгоор хөгжүүлсэн болно. Хэрэв values[i][j] байхгүй бол өөрөөр хэлбэл i дугаар мөрт харгарзах j дугаартай мөр байхгүй бол (эсрэгээрээ байж болно) 0 гэдэг утга буцааж байгаа болно.

setRow болон setColumn функцын хувьд матрицын хэмжээг өөрчлөөд гишүүдийн утгыг 0 болгож байгаа болно.

```
float Matrix::getValue(int i, int j) const
{
    if (i >= 1 && i <= row && j >= 1 && j <= column)
    {
        return values[i - 1][j - 1];
    }
    return 0; // values[i][j] ni matrix d baihgui bol 0 iig butsaana.
}

void Matrix::setValue(int i, int j, float x)
{
    if (i >= 1 && i <= row && j >= 1 && j <= column) // values[i][j] ni baihgu bol yu ch hiihgui.
        values[i - 1][j - 1] = x;
}

void Matrix::setRow(int newRow)
{
    if (newRow > 0)
    {
        float **newValues = new float *[newRow];
        for (int i = 0; i < newRow; i++)
        {
            newValues[i] = new float[column];
            for (int j = 0; j < column; j++)
            {
                newValues[i][j] = (i < row) ? values[i][j] : 0; // shine mor nemej ogj baigaa uchir shineer nemegdsen morud 0 utga awna
            }
        }
        // Umnuh sanah oigoo tseverleh heregtei double detected aldaa zaana.
        for (int i = 0; i < row; ++i)
            delete[] values[i];
        delete[] values;

        values = newValues; // shine utgig onooj ogno.
        row = newRow;       // shine moriig onoono.
    }
}

void Matrix::setColumn(int newColumn)
{
    // setColumn set Row toi ijil zarchimtai ajilna.
    if (newColumn > 0)
    {
        for (int i = 0; i < row; ++i)
        {
            float *newRow = new float[newColumn];
            for (int j = 0; j < newColumn; ++j)
            {
                newRow[j] = (j < column) ? values[i][j] : 0;
            }
            delete[] values[i];
            values[i] = newRow;
        }
        column = newColumn;
    }
}
```

4.4 +, - операторууд

Дээр дурдсанчлан + /нэмэх/ операторыг 2 янзаар тодорхойлсон ба нэгт матриц элемент бүрийг тогтмол тоогоор нэмэгдүүлэх, хоёрт хэмжээ нь адилхан 2 матрицийг хооронд нь нэмэх.

```
Matrix Matrix::operator+(float x) const
{
    Matrix temp(row, column); // temp nertei yag ijil matrix uusgene
    for (int i = 0; i < row; ++i)
    {
        for (int j = 0; j < column; ++j)
        {
            temp.values[i][j] = values[i][j] + x; // temp iin utga ni omnoh matrixin utga dr x utgig nemsentei tentsu
        }
    }
    return temp; // tuunig butsaana
}

Matrix Matrix::operator+(const Matrix &other) const
{
    if (row != other.row || column != other.column)
    {
        // Ug 2 matrixig nemeh bolomjgui bol ijil hernee null matrix butsaana
        return Matrix(row, column);
    }

    Matrix result(row, column); // hariultig uusgej ogno.
    for (int i = 0; i < row; ++i)
        for (int j = 0; j < column; ++j)
            result.values[i][j] = values[i][j] + other.values[i][j]; // ug matrixudin niilberig hadgalna
    return result;
}

Matrix Matrix::operator-(const Matrix &other) const
{
    // Ug uildel operator+ uildeltei ijil zarchimtai ajilna
    if (row != other.row || column != other.column)
    {
        return Matrix(row, column);
    }

    Matrix result(row, column);
    for (int i = 0; i < row; ++i)
        for (int j = 0; j < column; ++j)
            result.values[i][j] = values[i][j] - other.values[i][j];
    return result;
}
```

4.5 матрицыг үржүүлэх оператор

2 матрицыг үржүүлэхэд эхний матрицын баганы тоо 2 дох матрицын мөрийн тоотой тэнцүү байх хэрэгтэй. Энэ нөхцлийн шалгаад хэрэв уг 2 матриц хангахгүй бол эхний матрицийг буцааж байгаа болно.

```
Matrix Matrix::operator*(const Matrix &other) const
{
    if (column != other.row)
    {
        // ug 2 matrixig urjih bolomjgui bol 0 eer duursen matrix butsaana
        return Matrix(row, other.column);
    }

    Matrix result(row, other.column); // hariultig zarlana
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < other.column; j++)
        {
            for (int k = 0; k < column; k++)
            {
                result.values[i][j] += values[i][k] * other.values[k][j]; // matrixin urjih uildliig hiine
            }
        }
    }
    return result;
}
```

int Matrix::column

4.6 Матрицад нөгөө нэг матрицын утга оноох
Энэ үйлдлийг хийхэд 2 матрицын хэмжээ өөр байх боломжтой тул утга оноож байгаа матрицын хэмжээг 2 дох матрицын хэмжээнд тааруулан өөрчлөн хэрэгтэй

```
Matrix &Matrix::operator=(const Matrix &other)
{
    if (this != &other) // free(): double free detected erroros sergiilne
    {
        // Odoo baigaa sanah oig tseverleh
        for (int i = 0; i < row; i++)
        {
            delete[] values[i];
        }
        delete[] values;

        row = other.row;       // rowiin hemjeeg huulah
        column = other.column; // columiin hemjeeg huulah sanah oig zov ashigtai zarutsuulah heregtei
        values = new float *[row];
        for (int i = 0; i < row; i++)
        {
            values[i] = new float[column];
            for (int j = 0; j < column; j++)
                values[i][j] = other.values[i][j]; // huulah uildel
        }
    }
    return *this;
}
```

4.7 Нэгээр нэмэгдүүлэх болон хорогдуулах үйлдэл

```
Matrix &Matrix::operator++() // Matrixin utguudig 1eer nemegduuleh
{
    // Tuhain matrixin mor bolon baganaar gvih
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < column; j++)
        {
            values[i][j]++; // utgig 1 eer nemegduuleh
        }
    }
    return *this;
}

Matrix &Matrix::operator--() // Matrixin utguudig 1eer horogduulah
{
    // operator++ uildeltei yg ijil zarchimtai ajilna
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < column; j++)
        {
            values[i][j]--;
        }
    }
    return *this;
}
```

4.8 Матрицаар нэмэгдуулэх болон хорогдуулах үйлдэл
Хэрэв 2 матрицын хэмжээ өөр байвал юу ч хийхгүй байна. Хэрэв таарч байвал эхний матрицын гишүүн тус бүрийг нөгөө матрицын харгалзах гишүүнээр нэмэгдүүлэх болно.

```
Matrix &Matrix::operator+=(const Matrix &other) // Matrixig oor matrixaar nemegduuleh
{
    if (row == other.row && column == other.column) // Row bolon column tentsuu uyd nemegduulne
    {
        for (int i = 0; i < row; ++i)
        {
            for (int j = 0; j < column; ++j)
            {
                values[i][j] += other.values[i][j];
            }
        }
    }
    return *this; // Hervee mor bagana tentsuu bish uyd tuhain matrixig oorig ni butsaana esvel nemegdsen matrixig butsaana
}

Matrix &Matrix::operator-=(const Matrix &other) // Matrixig oor matrixaar horogduulah uildel
{
    // operator+= ijil zarchimtai ajilna
    if (row == other.row && column == other.column)
    {
        for (int i = 0; i < row; ++i)
        {
            for (int j = 0; j < column; ++j)
            {
                values[i][j] -= other.values[i][j];
            }
        }
    }
    return *this;
}
```

4.9 Нэг матрицыг нөгөө матрицаар үржүүлэх үйлдэл
Энэ үйлдэл хийхэд 2 матриц үржүүлж байгаатай төстэй үйлдэл хийх юм. 2 матрицыг үржүүлэхэд эхний матрицын баганы тоо 2 дох матрицын мөрийн тоотой тэнцүү байх хэрэгтэй. Энэ нөхцлийн шалгаад хэрэв уг 2 матриц хангахгүй бол юу ч хийхгүй болно. Хэрэв хангаж байгаа бол эхний матрицын хэмжээг өөрчлөх хэрэгтэй болно.

```
Matrix &Matrix::operator*=(const Matrix &other) // Matrixig oor negen matrixaar urjuuleh uildel
{
    if (column == other.row)
    {
        Matrix temp(*this * other);
        *this = temp;
    }
    return *this;
}
```

4.10 Матриц хөрвүүлэх

Уг матрицын хэмжээг өөрчлөх хэрэгтэй.

```
void Matrix::convert() // Matrixig horwuuleh uildel
{
    Matrix temp(*this);           // ijil hemjeetei shine matrix zarlana
    for (int i = 0; i < row; ++i) // odoo baigaa matrixin sanah oig choloolno
    {
        delete[] values[i];
    }
    delete[] values;

    row = temp.column; // horwuulj bui uchir row bolon columnig solino
    column = temp.row;
    values = new float *[row];
    for (int i = 0; i < row; ++i)
    {
        values[i] = new float[column];
        for (int j = 0; j < column; ++j)
        {
            values[i][j] = temp.values[j][i]; // solison row columnin utguudig hoorond ni solino
        }
    }
}
```

5. ДҮГНЭЛТ

C++ хэлний классын хүрээнд оператор дахин тодорхойлох нь хөгжүүлэгчийн бичсэн коддыг уншууртай нөгөө талаас бага коддоор их үйлдэл хийх боломжийг олгодог давуу талуудтай. Матриц дээр хийх үйлдэл зэрэг арифметик үйлдлүүдийг олон удаа давтан хийх шаардлагатай байгаа тохиолдолд уг операторыг классын түвшинд нэг удаа тодорхойлон олон удаа ашиглах боломжтой юм.

АШИГЛАСАН МАТЕРИАЛ

(2023, November 2). Retrieved from geeksforgeeks: https://www.geeksforgeeks.org/operator-overloading-cpp/

6. Хавсралт

https://github.com/NaraaDev/NUM-OBJECT // Дасгал ажлын код