객체지향프로그래밍

Assignment 5

학과: 수학과

학번: 20140843

이름: 엄태강

POVIS ID: tkeom0114

Honor Code:나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.

1. 개요

이번 과제에서는 QT환경에서 간단한 GUI프로그램인 계산기 프로그램을 만들었다. 또한 Operator overloading을 이용해 행렬, 문자열에 대해서도 계산할 수 있도록 만들었다.

2.class 구성

이번 과제에서는 계산할 data type을 Operand를 base class로 둬서 행렬(Matrix), 실수(Numeric), 문자열(StrValue)를 class로 구현했다.

base class인 Operand는 아래와 같이 구성되어 있다. 이 class는 abstract base class이다.

class Operand

{

public:

Operand(QString,QString);

Operand();

QString getName()const;

void setName(QString);

QString getType();

virtual ~*Operand*();

virtual void *printValue*(QTextBrowser\*)=0;//text browser에 변수의 내용을 출력하는 함수로 Operand에서는 pure virtual function이다.

private:

QString name;//해당 변수의 이름이다

QString type;//Matrix,Numeric,String중 어떤 타입인지 나타낸다.

};

class Matrix : public Operand

{

public:

Matrix(QString,int,int,double);

Matrix();

~Matrix();

virtual void printValue(QTextBrowser\*);//text browser에 변수의 내용을 출력하는 함수이다.

int getCols()const;

int getRows()const;

double getValue(int,int)const;

bool setValue(int,int,double);

friend Matrix operator+(const Matrix&,const Matrix&);

friend Matrix operator-(const Matrix&,const Matrix&);

friend Matrix operator\*(const Matrix&,const Matrix&);

friend Matrix operator+(const Matrix&,const Numeric&);

friend Matrix operator-(const Matrix&,const Numeric&);

friend Matrix operator\*(const Matrix&,const Numeric&);

friend Matrix operator/(const Matrix&,const Numeric&);

friend Matrix operator+(const Numeric&,const Matrix&);

friend Matrix operator-(const Numeric&,const Matrix&);

friend Matrix operator\*(const Numeric&,const Matrix&);

friend Matrix operator/(const Numeric&,const Matrix&);

private:

int cols;

int rows;

double \*\*mat;

};

class Numeric : public Operand

{

public:

Numeric(QString,double);

Numeric();

~Numeric();

virtual void printValue(QTextBrowser\*);//text browser에 변수의 내용을 출력하는 함수이다.

double getValue()const;

void setValue(double);

//아래의 함수들은 top-level operate overloading 함수들이다.

friend Numeric operator+(const Numeric&,const Numeric&);

friend Numeric operator-(const Numeric&,const Numeric&);

friend Numeric operator\*(const Numeric&,const Numeric&);

friend Numeric operator/(const Numeric&,const Numeric&);

friend Matrix operator+(const Matrix&,const Numeric&);

friend Matrix operator-(const Matrix&,const Numeric&);

friend Matrix operator\*(const Matrix&,const Numeric&);

friend Matrix operator/(const Matrix&,const Numeric&);

friend Matrix operator+(const Numeric&,const Matrix&);

friend Matrix operator-(const Numeric&,const Matrix&);

friend Matrix operator\*(const Numeric&,const Matrix&);

friend Matrix operator/(const Numeric&,const Matrix&);

friend StrValue operator+(const StrValue &,const Numeric &);

friend StrValue operator\*(const StrValue &,const Numeric &);

friend StrValue operator+(const Numeric &,const StrValue &);

friend StrValue operator\*(const Numeric &,const StrValue &);

private:

double value;

};

class StrValue : public Operand

{

public:

StrValue(QString,QString);

StrValue();

~StrValue();

virtual void printValue(QTextBrowser\*);

QString getValue()const;

void setValue(QString);

//아래의 함수들은 top-level operate overloading 함수들이다.

friend StrValue operator+(const StrValue &,const StrValue &);

friend StrValue operator+(const StrValue &,const Numeric &);

friend StrValue operator\*(const StrValue &,const Numeric &);

friend StrValue operator+(const Numeric &,const StrValue &);

friend StrValue operator\*(const Numeric &,const StrValue &);

private:

QString value;

};

3.알고리즘

1) main(mainwindow.c,main.c)

①list <Operand\*> Operands를 만들어 변수들을 저장할 수 있도록 한다.

②Text browser과 button들을 띄워 계산기를 화면에 보여준다.

2)변수 출력(void MainWindow::printAllValues (), void printValue (QTextBrowser\*))

①list <Operand\*> Operands를 순회하며 변수 타입에 따라 서로 다른 printValue를 호출한다.

②printValue에서는 변수의 이름, 내용을 출력한다.

3)변수 추가(adddialog.cpp)

①Matrix, Numeric, String중 한 버튼을 클릭하면 변수 추가를 위한 dialog object를 만든다.

②dialog의 line edit에 추가할 변수의 정보를 입력할 수 있다.

③설정이 잘못된 경우(예를 들면 이름이 “result") OK를 누르면 message box를 띄워 error가 났음을 알린다.

④설정이 올바른 경우 OK를 누르면 변수를 생성하고, mainwindow의 textbrowser에 signal을 보내 printAllVaule를 호출하고, 생성된 dialog를 닫는다.

4)변수 편집(editdialog.cpp)

①Edit Value 버튼을 클릭하면 변수 추가를 위한 dialog object를 만든다.

②dialog의 combobox에 Operands에 저장된 변수들의 이름을 띄우고 선택된 변수의 타입을 combobox 아래의 label에 출력한다.

③ dialog의 line edit에 변수의 정보를 입력할 수 있다.

④OK를 누르면 변수를 입력한 대로 편집하고 mainwindow의 textbrowser에 signal을 보내 printAllVaule를 호출하고, 생성된 dialog를 닫는다. textbrowser를 통해 변수가 편집되었음을 확인할 수 있다.

5)변수 삭제(operator.cpp)

①Delete Variable 버튼을 클릭하면 변수 추가를 위한 dialog object를 만든다.

②dialog의 combobox에 Operands에 저장된 변수들의 이름을 띄우고 선택된 변수의 타입을 combobox 아래의 label에 출력한다.

③OK를 누르면 Operands에서 변수를 제거하고, 생성된 dialog를 닫는다.

6)연산(operator.cpp)

①+,-,x,/중 한 버튼을 클릭하면 연산 위한 dialog object를 만든다.

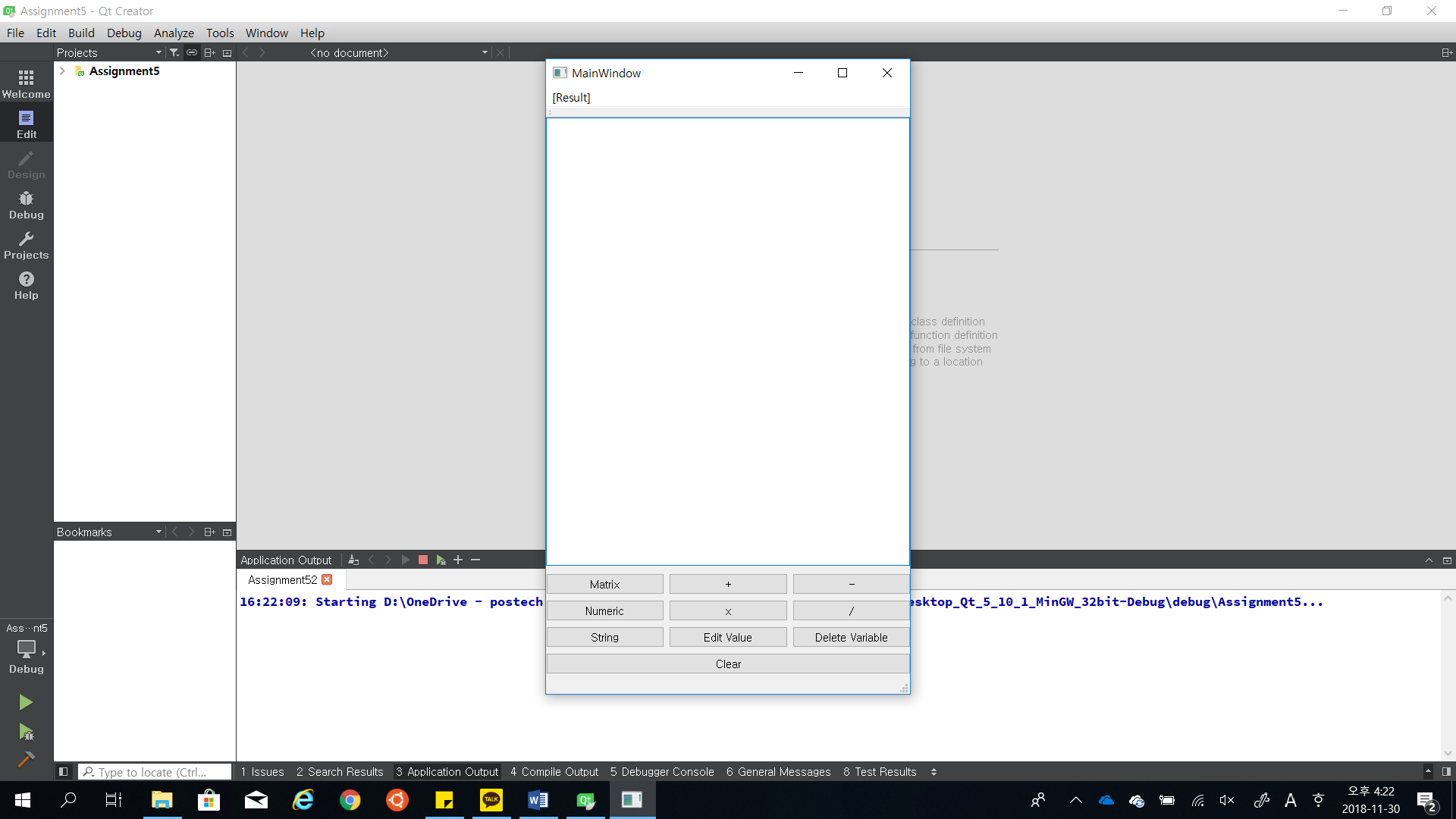
② dialog의 2개의 combobox에 Operands에 저장된 변수들의 이름을 띄우고 선택된 변수의 타입을 combobox 아래의 label에 출력한다.

③잘못된 변수들을 선택한 경우 (예를 들면 matrix와 string간의 계산) OK를 누르면 message box를 띄워 error가 났음을 알린다.

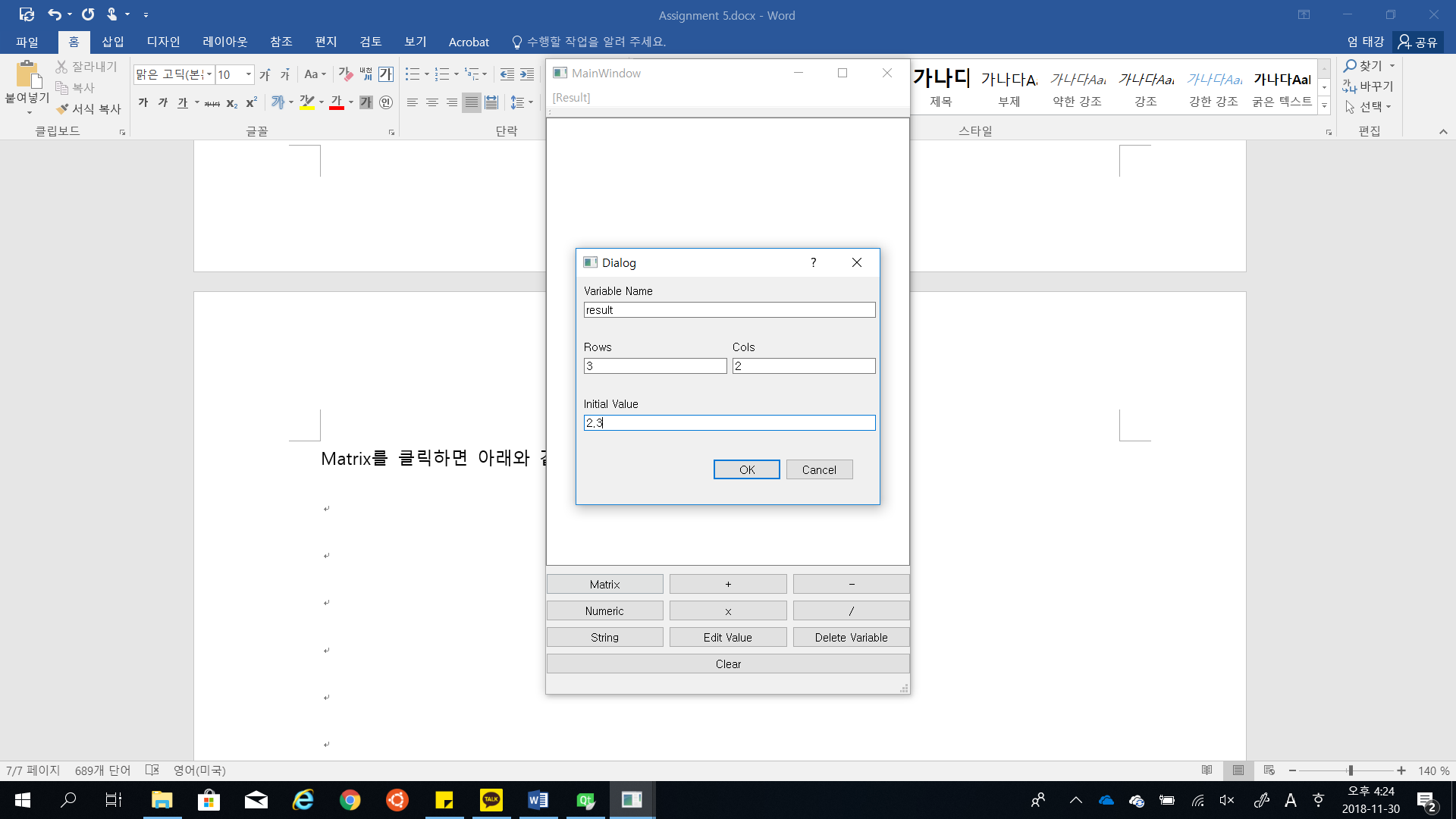
④설정이 올바른 경우 OK를 누르면 계산 결과에 따른 변수(result)를 생성하고, mainwindow의 textbrowser에 signal을 보내 printAllVaule를 호출하고, 생성된 dialog를 닫는다.

4.실행방법&실행예시

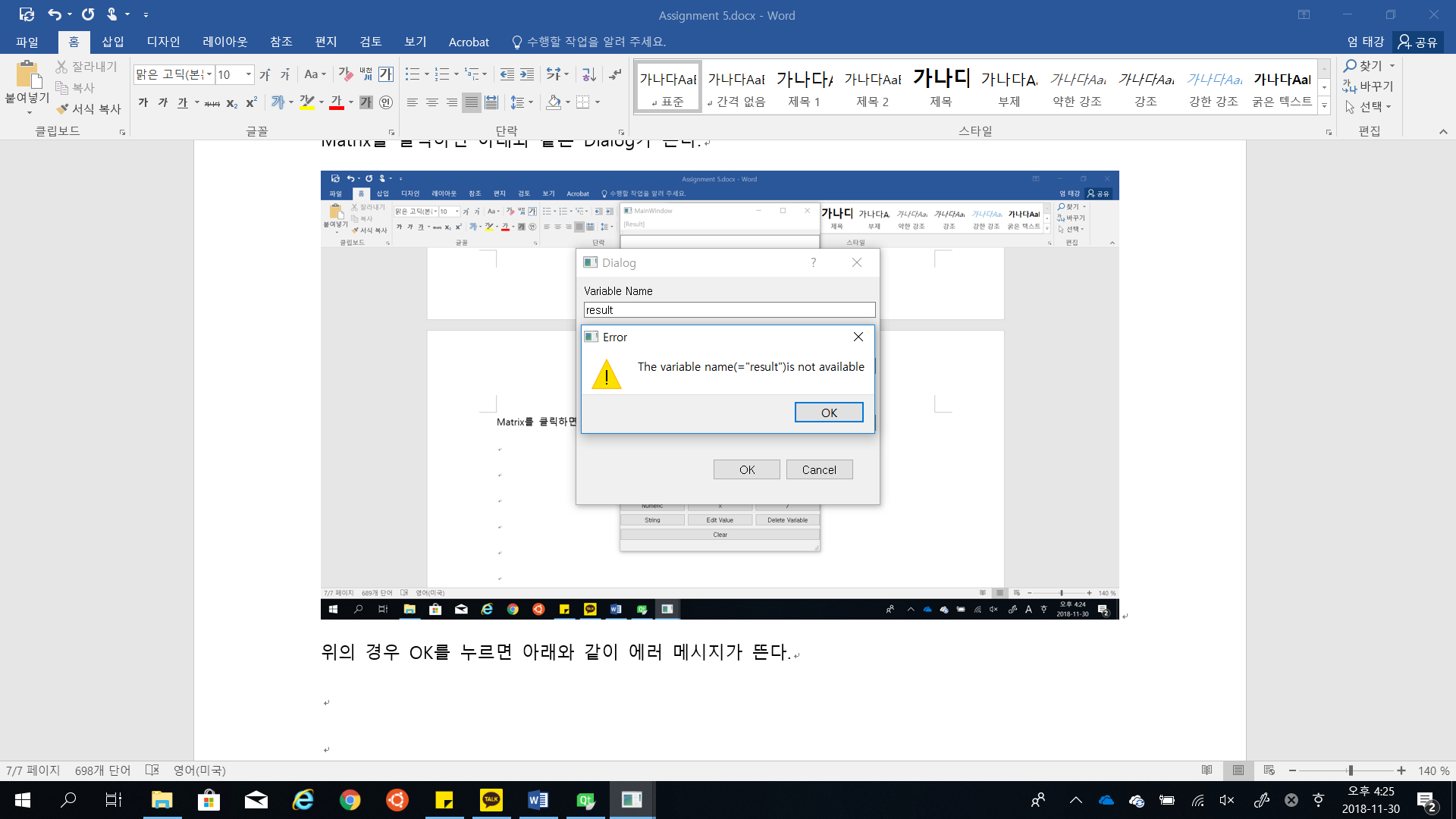
Qt creator에서 build를 하면 아래와 같이 계산기 프로그램이 실행된다.



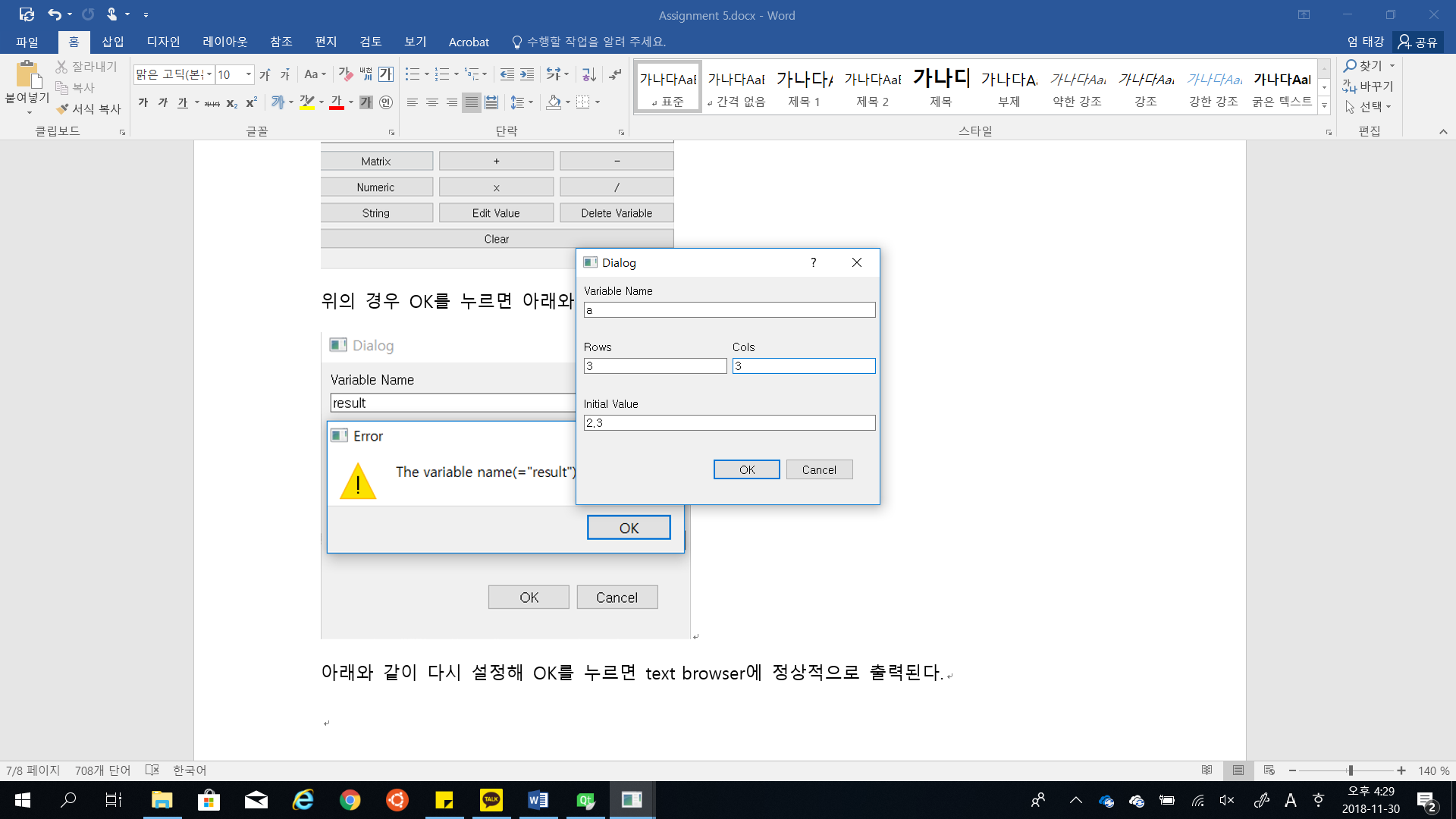
Matrix를 클릭하면 아래와 같은 Dialog가 뜬다.

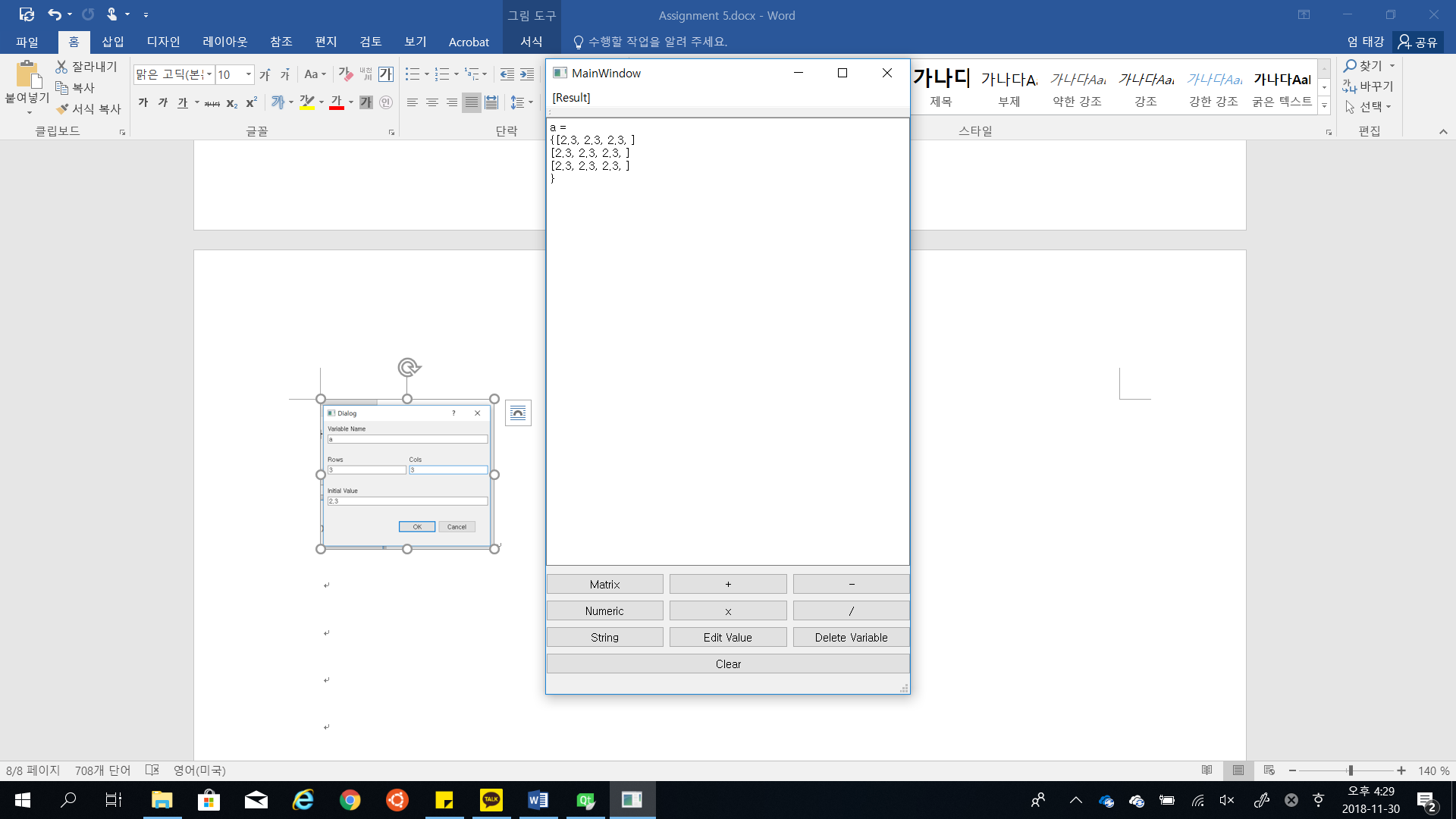


위의 경우 OK를 누르면 아래와 같이 에러 메시지가 뜬다.

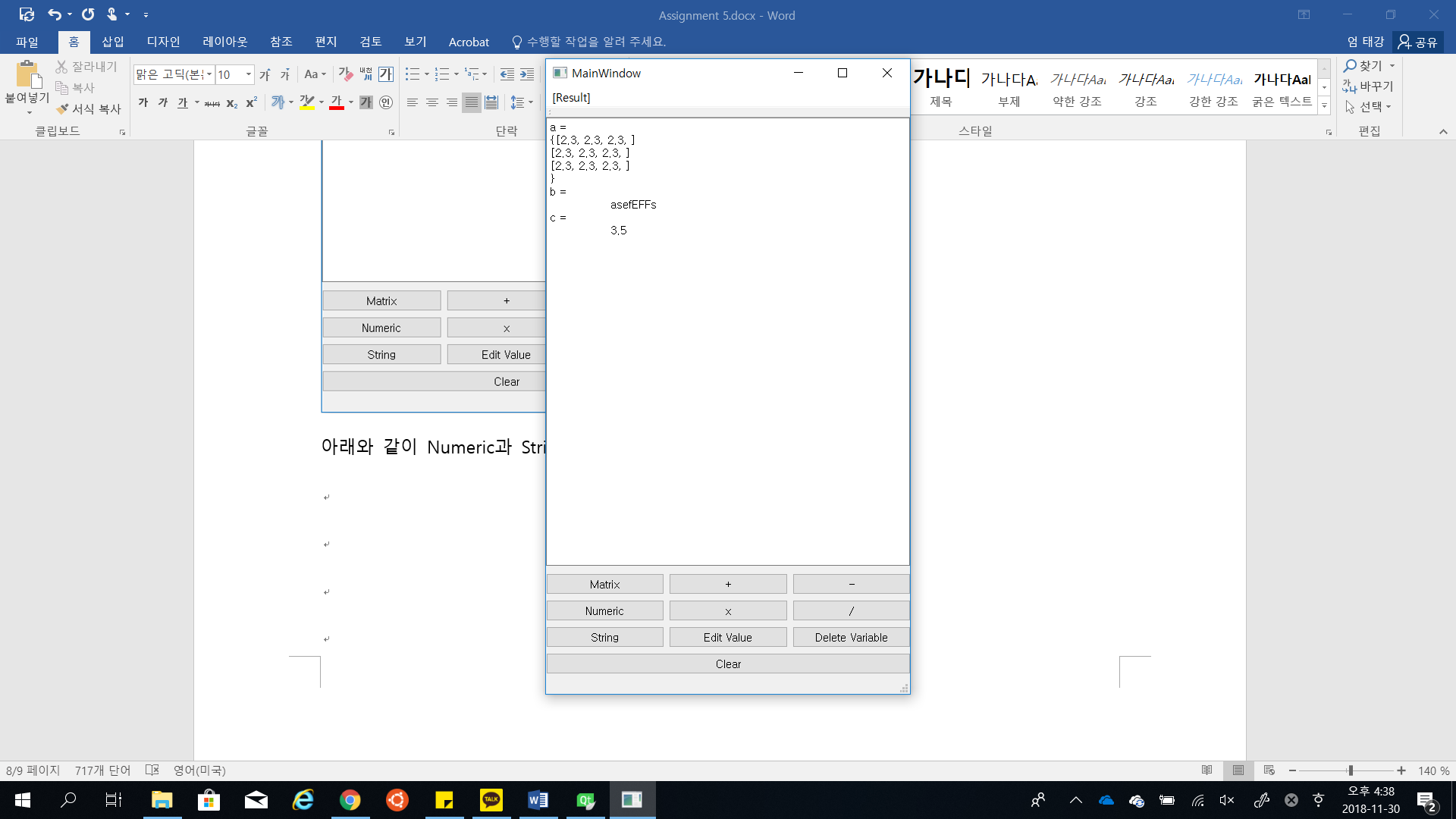


아래와 같이 다시 설정해 OK를 누르면 text browser에 정상적으로 출력된다.

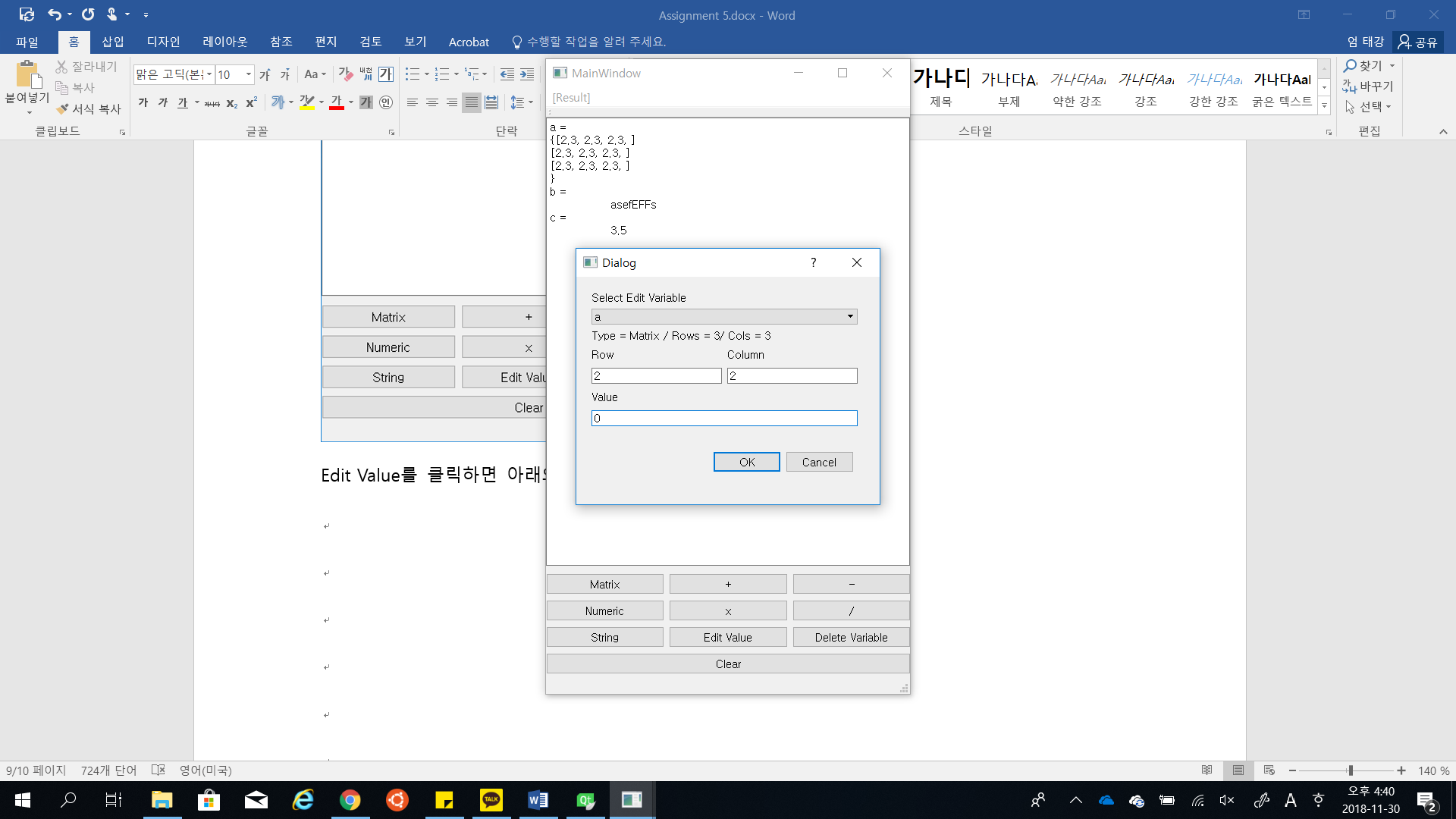




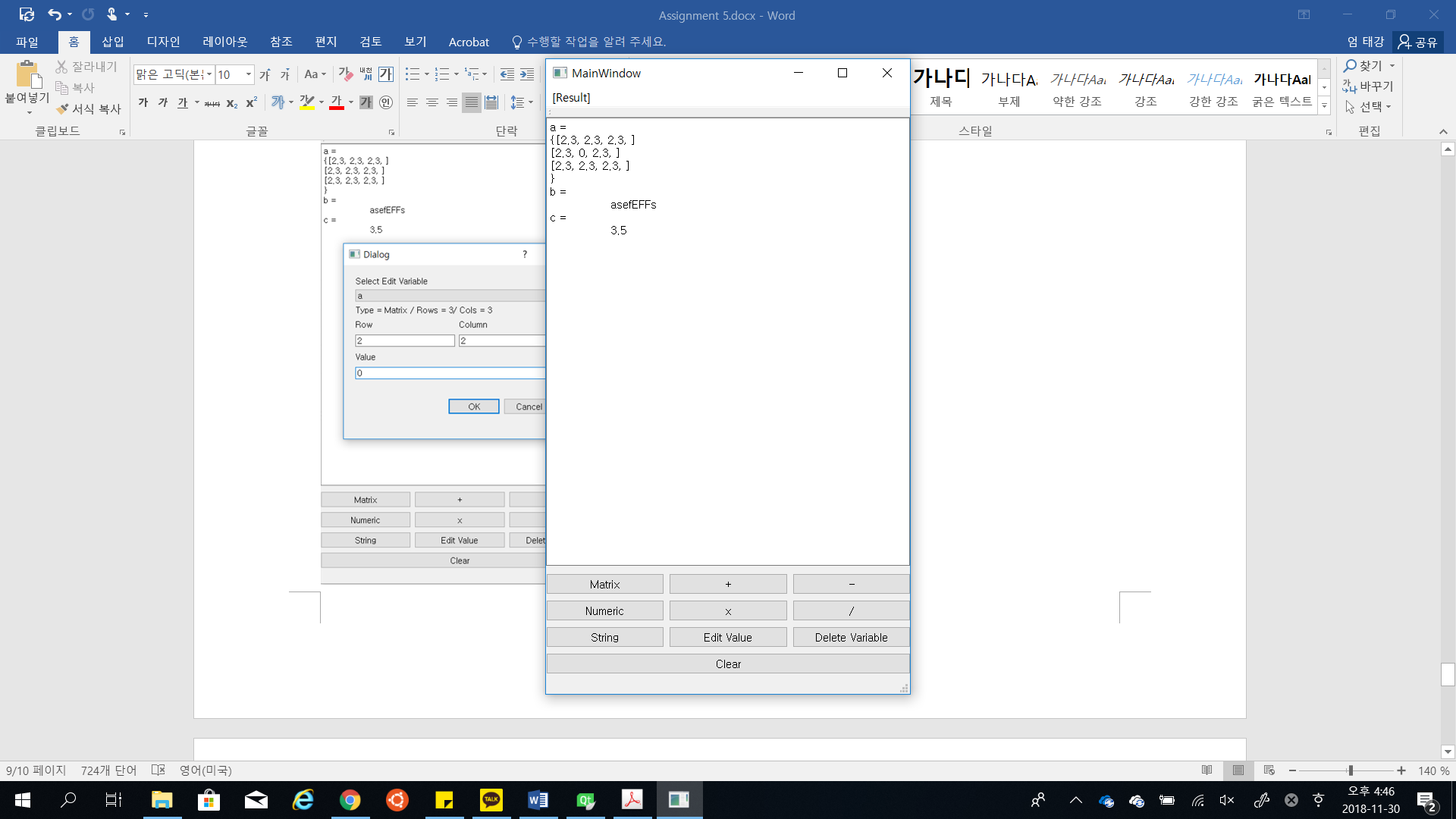
아래와 같이 Numeric과 String도 정상적으로 저장됨을 알 수 있다.



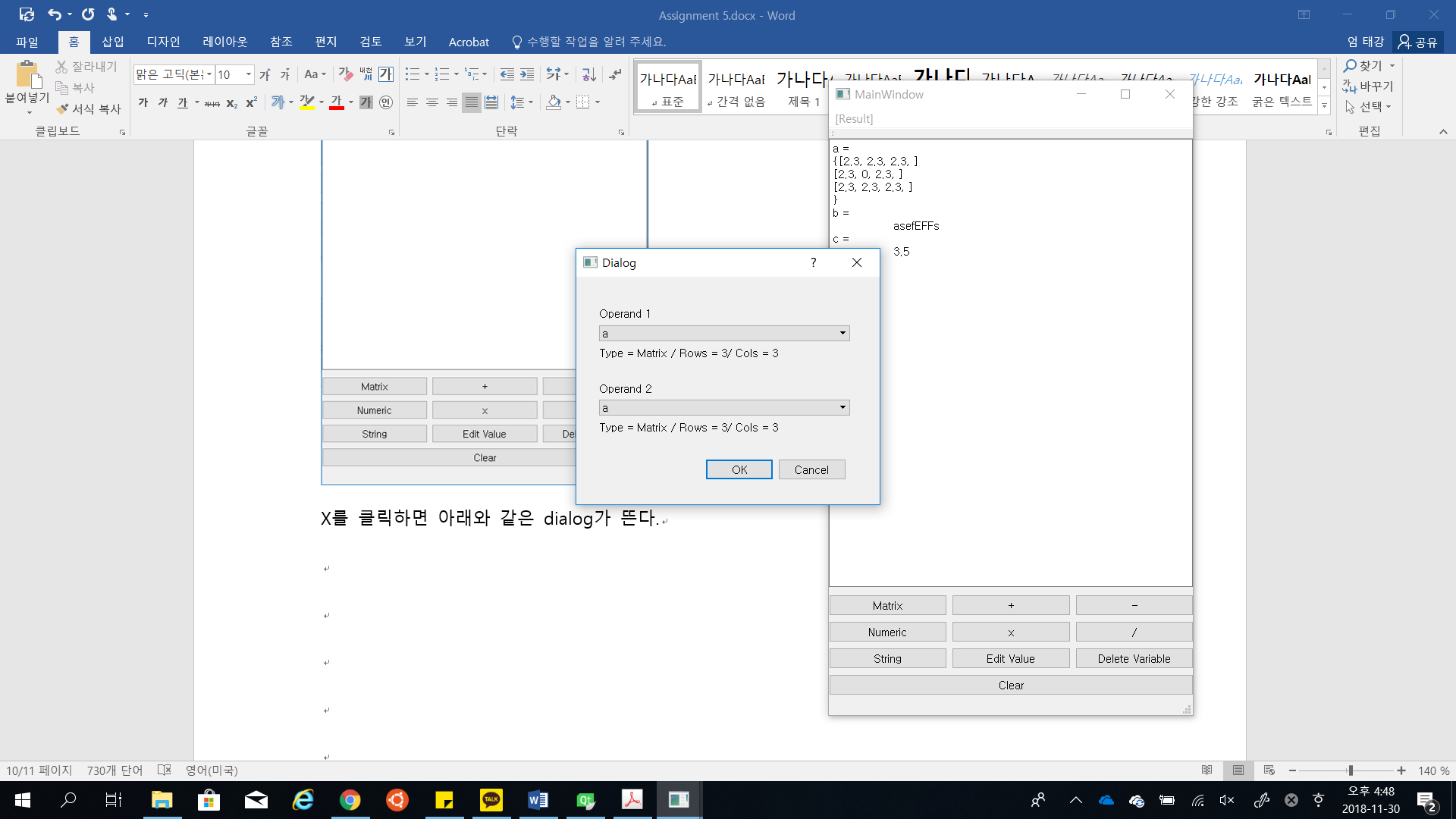
Edit Value를 클릭하면 아래와 같은 dialog가 뜬다.



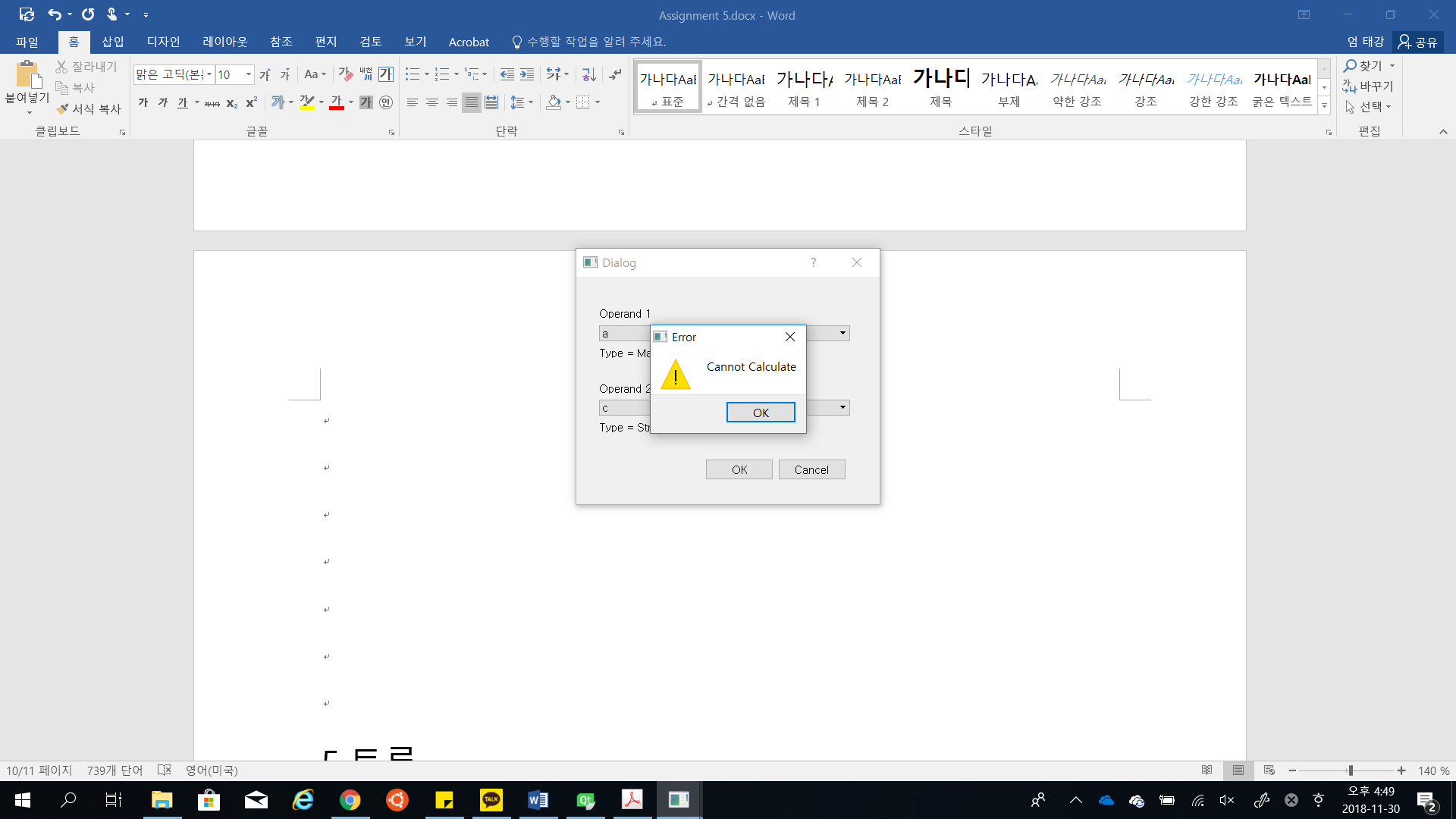
OK를 클릭하면 아래와 같이 변수가 수정된다.



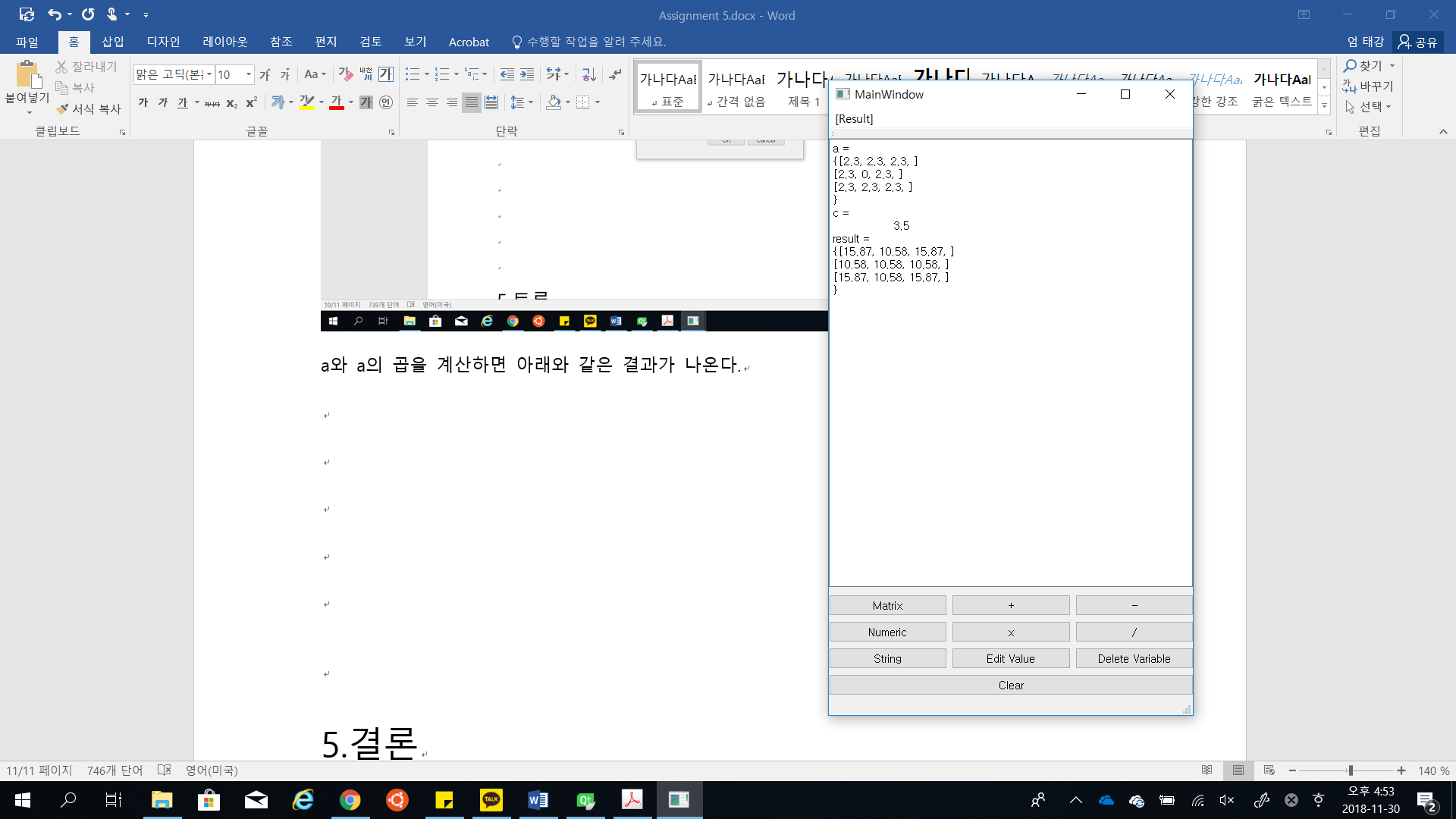
X를 클릭하면 아래와 같은 dialog가 뜬다.



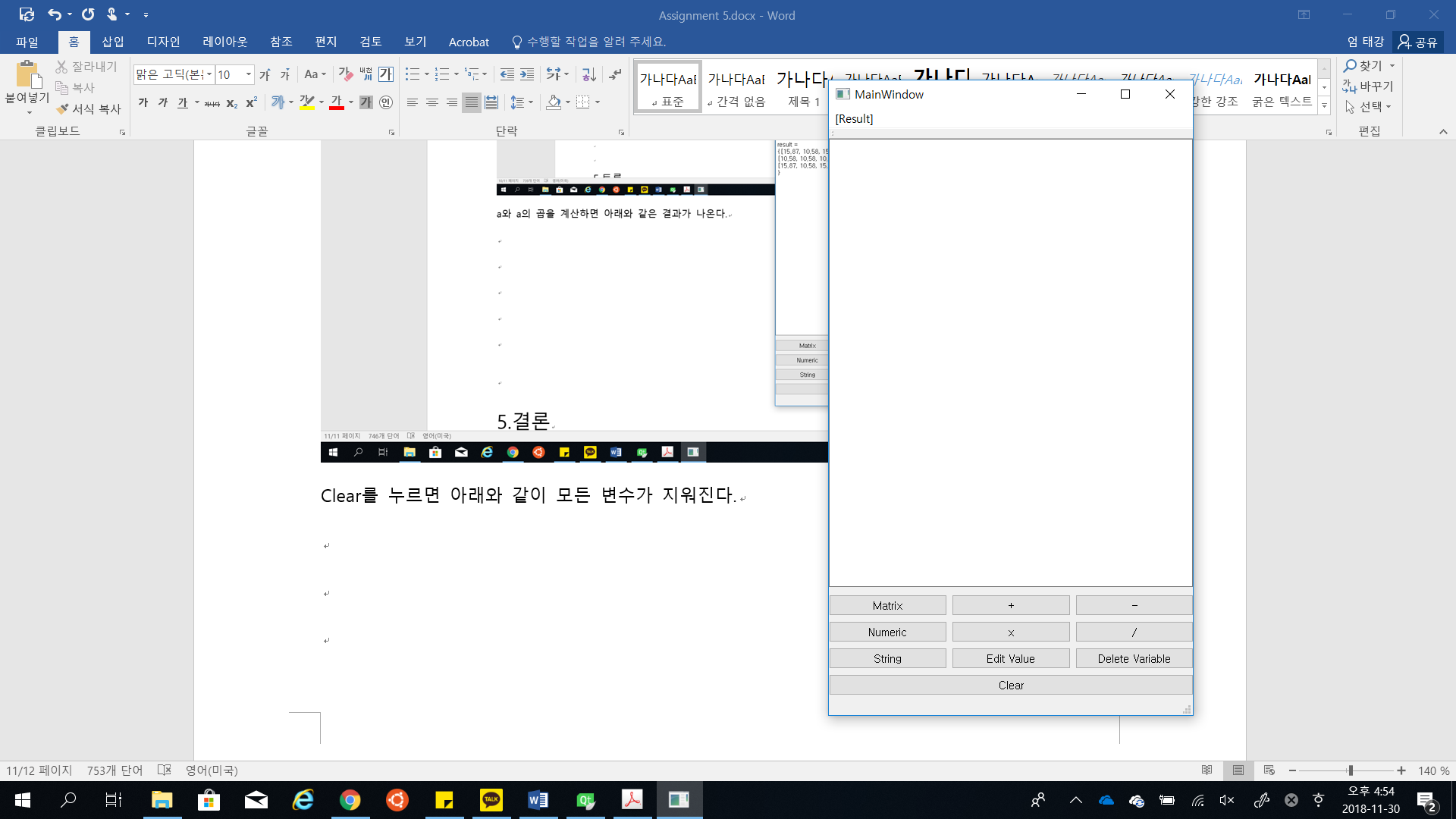
matrix와 string을 연산하려고 하면 아래와 같이 에러 메시지를 띄운다.



a와 a의 곱을 계산하면 아래와 같은 결과가 나온다.



Clear를 누르면 아래와 같이 모든 변수가 지워진다.



5.토론

Edit Value에서 matrix의 entry를 입력할 때 사용자 입장에서 생각해 row=1,col=1을 넣을 때, 첫 번째 행과, 첫 번째 열의 entry를 수정하도록 한다.

6.결론

이번 과제에서는 QT환경에서 간단한 GUI프로그램인 계산기 프로그램을 만들었다. QT사용법이 익숙하지 않아 간단한 프로그램임에도 불구하고 많은 시행착오가 있었다. 하지만 QT를 직접 사용해봄으로써 프로젝트를 할 때 좀 더 쉽게 QT를 사용할 수 있을 것 같다.