

C++_예외처리, 파일입출력, 파일 배포


Visual Studio 2022

예외 처리(Exception Handling)

프로그램 실행 중에 예외가 발생한 경우에 대한 처리 과정
예외처리가 없는 경우 정상적인 프로그램 실행이 안 될 수도 있다.

try ~ catch 구문을 이용하고, throw 키워드 사용함

```
try
{
    정상적인 처리 내용
    예외 발생 경우 throw 전달인수;
}
catch(throw에서 전달받은 인수)
{
    예외 발생시 수행할 내용
}
```



예외 처리(Exception Handling)

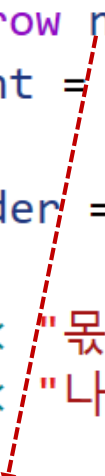
나누기 연산에서 0으로 나누었을때(분모가 0인 경우) 예외 처리

```
int n1, n2;  
int quotient, remainder;  
  
cout << "첫번째 수 입력: ";  
cin >> n1;  
  
cout << "두번째 수 입력: ";  
cin >> n2;
```

```
첫 번째 수 입력 : 10  
두 번째 수 입력 : 0  
10은 0으로 나눌 수 없습니다.
```

try~catch 구문

```
try {  
    if (n2 == 0)  
        //cout << n1 << "는(은) 0으로 나눌 수 없습니다\n";  
        throw n1; //catch()의 매개변수로 보냄  
    quotient = n1 / (double)n2; //몫 계산  
  
    remainder = n1 % n2;           //나머지 계산  
  
    cout << "몫: " << quotient << endl;  
    cout << "나머지: " << remainder << endl;  
}  
catch (int e_n) {  
    cout << n1 << "는(은) 0으로 나눌 수 없습니다.\n";  
}
```



함수에서 예외 블록 호출하기

문자열을 정수로 변환하는 stringToInt 함수를 구현하고,
변환 중에 숫자가 아닌 문자가 발견되면 예외를 발생시키는 프로그램

```
/*char s[] = "apple";
|   cout << strlen(s) << endl;*/

/*int x = '0';
int y = '1';
cout << x << endl; //아스키 코드값 48
cout << y << endl; //아스키 코드값 49
cout << 1 - 0 << endl;
cout << '1' - '0' << endl;
*/|
```

함수에서 예외 블록 호출하기

- 문자열을 정수로 변환하는 프로그램

```
#include <iostream>
#include <cstring> //strlen() 사용
using namespace std;

/*
 * 문자열을 정수로 변환하는 프로그램
 */
int stringToInt(const char x[]) {
    int sum = 0;
    int len = strlen(x);
    for (int i = 0; i < len; i++) {
        if (x[i] > '0' && x[i] < '9')
            sum = sum * 10 + x[i] - '0';
        else
            throw x; //예외를 catch의 인자로 보냄
    }
    return sum;
}
```

try~catch 구문

- 문자열을 정수로 변환하는 프로그램

```
int main()
{
    int n;
    //예외 처리 : try ~ catch 구문
    try {
        /*n = stringToInt("12");
        cout << "\"12\" 는 정수 " << n << "으로 변환됨\n";*/

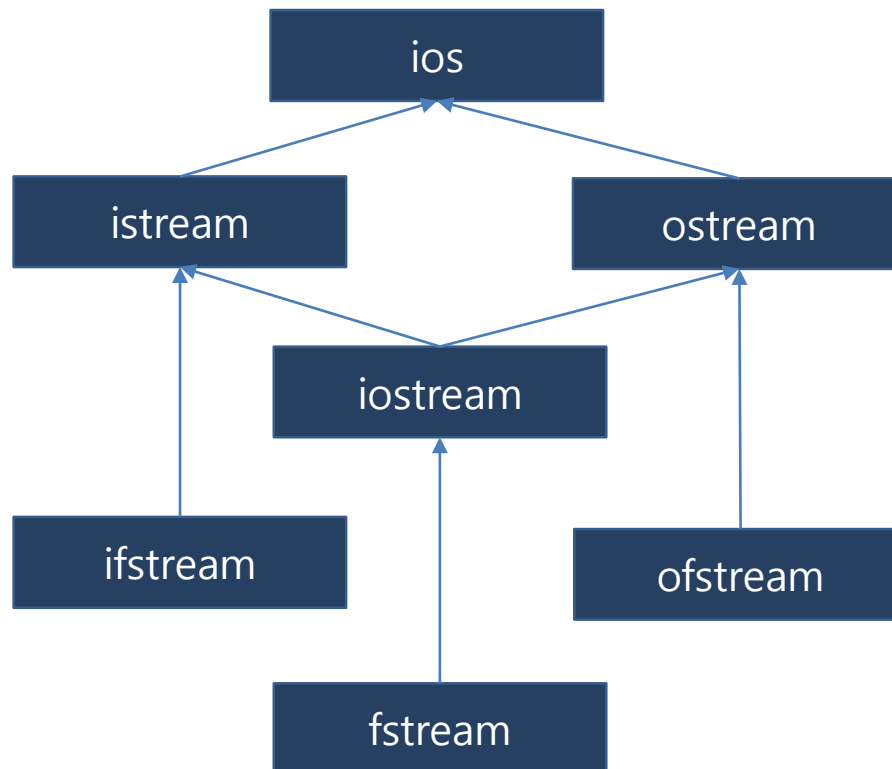
        n = stringToInt("12a");
        cout << "\"12a\" 는 정수 " << n << "으로 변환됨\n";
    }
    catch (const char* str) {
        cout << str << " 처리에서 예외 발생!" << endl;
    }

    return 0;
}
```

파일 입출력

➤ 파일 입출력의 필요성

프로그램 실행 중에 메모리에 저장된 데이터는 프로그램이 종료되면 사라진다. 데이터를 프로그램이 종료된 후에도 계속해서 사용하려면 파일에 저장하고 필요할때 파일을 읽어서 데이터를 사용할 수 있다.



파일 입출력

```
#include <fstream>
```

```
ofstream f1("output.txt"); //출력 객체 생성(파일 열기)
```

```
if(f1.fail()) // 파일이 없을때 종료
```

```
    return 1;
```

```
    f1 << 내용 << endl; //파일에 쓰기
```

```
    out.close() //파일 닫기
```

파일 쓰기

```
#include <fstream>
```

```
ifstream f1("output.txt"); //입력 객체 생성(파일열기)
```

```
if(f1.fail())
```

```
    return 1;
```

```
while(!f1.eof()){ //파일의 끝까지 읽기 }
```

```
in.close() //파일 닫기
```

파일 읽기

파일 입출력 스트림

파일 쓰기

```
#include <iostream>
#include <fstream> //ofstream 사용
using namespace std;

int main()
{
    ofstream f1("data.txt"); // 파일 쓰기 객체 생성
    int x = 1, y = 2;

    if (!f1) {
        cerr << "파일을 열 수 없습니다.\n";
        return 1; // 오류 코드 반환
    }
}
```

파일 입출력 스트림

파일 쓰기

```
// 파일에 쓰기  
f1 << x << " " << y << endl;  
f1 << "Good Job!";  
  
f1.close(); // 파일 닫기  
  
return 0;  
}
```

data.txt

파일	편집	보기
----	----	----

1 2		
Good Job!		

파일 입출력 스트림

파일 읽기

```
#include <iostream>
#include <fstream> //ifstream 사용
#include <string>

using namespace std;

int main()
{
    ifstream f1("data1.txt"); //f1 객체 생성
    if (f1.fail()) {
        cerr << "파일을 찾을 수 없습니다.\n";
        return 1;
    }
}
```

파일 입출력 스트림

파일 읽기

```
string str; //읽은 내용 저장할 변수
/*while (!f1.eof()) { //end of file
    getline(f1, str);
    cout << str << endl;
}*/

while (getline(f1, str)) {
    cout << str << endl;
}

f1.close(); //파일 닫기

return 0;
}
```

```
1 2
Good Job!
```

파일 입출력 스트림

● 구구단 파일에 쓰기

파일	편집	보기
$2 \times 1 = 2$		
$2 \times 2 = 4$		
$2 \times 3 = 6$		
$2 \times 4 = 8$		
$2 \times 5 = 10$		
$2 \times 6 = 12$		
$2 \times 7 = 14$		
$2 \times 8 = 16$		
$2 \times 9 = 18$		
$3 \times 1 = 3$		
$3 \times 2 = 6$		
$3 \times 3 = 9$		
$3 \times 4 = 12$		
$3 \times 5 = 15$		
$3 \times 6 = 18$		
$3 \times 7 = 21$		
$3 \times 8 = 24$		
$3 \times 9 = 27$		

$8 \times 1 = 8$
 $8 \times 2 = 16$
 $8 \times 3 = 24$
 $8 \times 4 = 32$
 $8 \times 5 = 40$
 $8 \times 6 = 48$
 $8 \times 7 = 56$
 $8 \times 8 = 64$
 $8 \times 9 = 72$

$9 \times 1 = 9$
 $9 \times 2 = 18$
 $9 \times 3 = 27$
 $9 \times 4 = 36$
 $9 \times 5 = 45$
 $9 \times 6 = 54$
 $9 \times 7 = 63$
 $9 \times 8 = 72$
 $9 \times 9 = 81$

파일 입출력 스트림

● 구구단 파일에 쓰기

```
ofstream f1("gugudan.txt"); // 파일 쓰기 객체 생성

if (!f1) {
    cerr << "파일을 열 수 없습니다.\n";
    return 1; // 오류 코드 반환
}

// 구구단 쓰기
for (int i = 2; i <= 9; i++) {
    for (int j = 1; j <= 9; j++) {
        f1 << i << " x " << j << " = " << (i * j) << endl;
    }
    f1 << endl;
}

f1.close(); // 파일 닫기
```

파일 입출력 스트림

● 구구단 읽기

```
ifstream f1("gugudan.txt"); // 파일 객체 생성
string str;

if (!f1) {
    cerr << "파일을 열 수 없습니다.\n";
    return 1; // 오류 코드 반환
}

// 구구단 읽기
while (getline(f1, str)) {
    cout << str << endl;
}

f1.close(); // 파일 닫기
```

```
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18

3 x 1 = 3
3 x 2 = 6
3 x 3 = 9
3 x 4 = 12
3 x 5 = 15
3 x 6 = 18
3 x 7 = 21
3 x 8 = 24
3 x 9 = 27
```


파일에 성적 저장하기

● 성적 저장하기

```
ofstream out("score.txt"); //파일 객체 생성
if (!out) {
    cerr << "Error: 파일을 열 수 없습니다.\n";
    return 1;
}

string name;    //이름
int eng, math;  //영어, 수학
cout << "이름 입력: ";
cin >> name;
cout << "영어점수 입력: ";
cin >> eng;
cout << "수학점수 입력: ";
cin >> math;

// 파일에 쓰기
out << name << " " << eng << " " << math << endl;

out.close(); //파일 닫기
cout << "데이터가 성공적으로 저장되었습니다!" << endl;
```

파일에 여러명 성적 저장하기

- 성적 리스트 만들기

```
1번째 학생의 이름 : 이정후  
영어점수 입력 : 87  
수학점수 입력 : 88  
2번째 학생의 이름 : 최민정  
영어점수 입력 : 91  
수학점수 입력 : 86  
3번째 학생의 이름 : 신유빈  
영어점수 입력 : 87  
수학점수 입력 : 76  
데이터가 성공적으로 저장되었습니다!
```

파일에 여러명 성적 저장하기

- 성적 리스트 만들기

```
class Student {  
private:  
    string name;    //이름  
    int eng;        //영어 점수  
    int math;       //수학 점수  
    double avg;     //평균  
  
public:  
    // 설정자(setter) 메서드들  
    void setName(string name) { this->name = name;}  
    void setEng(int eng) { this->eng = eng;}  
    void setMath(int math) { this->math = math;}  
  
    // 평균 계산 메서드  
    void calculateAvg() {  
        avg = (double)(eng + math) / 2;  
    }  
}
```

파일에 여러명 성적 저장하기

● 성적 리스트 만들기

```
// 접근자(getter) 메서드들
string getName() const { return name; }
int getEng() const { return eng; }
int getMath() const { return math; }
double getAvg() const { return avg; }
};

int main() {
    ofstream out("scorelist.txt");
    Student students[3];

    if (!out) {
        cerr << "Error: 파일을 열 수 없습니다.\n";
        return 1;
    }

    for (int i = 0; i < 3; i++) {
        string name;
        int eng, math;

        cout << i + 1 << "번째 학생의 이름: ";
        cin >> name;
        students[i].setName(name);
    }
}
```

파일에 여러명 성적 저장하기

● 성적 리스트 만들기

```
cout << "영어점수 입력: ";
cin >> eng;
students[i].setEng(eng);

cout << "수학점수 입력: ";
cin >> math;
students[i].setMath(math);

students[i].calculateAvg(); //평균 계산
}

// 파일에 쓰기
for (int i = 0; i < 3; i++) {
    out << students[i].getName() << " "
        << students[i].getEng() << " "
        << students[i].getMath() << " "
        << students[i].getAvg() << endl;
}

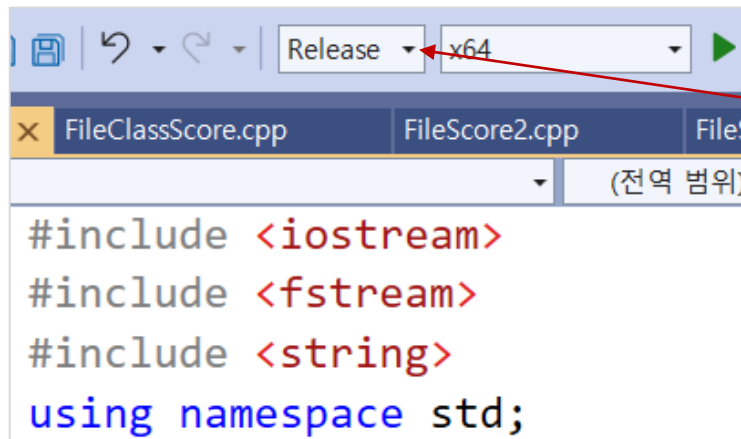
out.close();
cout << "데이터가 성공적으로 저장되었습니다!" << endl;
return 0;
}
```

이정후 87 88 87.5
최민정 91 86 88.5
신유빈 87 76 81.5

파일 배포

◆ 파일 배포

파일 배포란 c++언어 소스파일을 .exe 실행 파일로 만들어 공개 및 서비스 하는 것을 말한다.



Debug 모드를
Release 모드로 바꾼다.



Ctrl + F5로 실행

파일 배포

◆ exe 파일이 실행되지 않는 문제 해결

system("pause") 를 명시함

```
// 파일에 쓰기
for (int i = 0; i < 3; i++) {
    out << students[i].getName() << " "
        << students[i].getEng() << " "
        << students[i].getMath() << " "
        << students[i].getAvg() << endl;
}

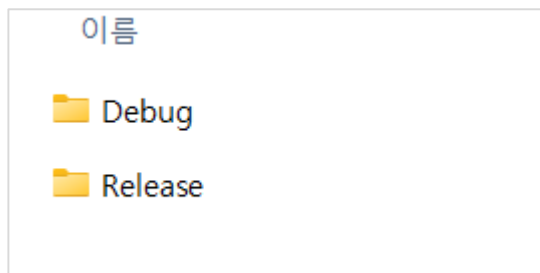
out.close();
cout << "데이터가 성공적으로 저장되었습니다!" << endl;

system("pause");

return 0;
```

파일 배포

◆ 파일 실행하기



Release 폴더 생성됨



FileIO.exe 파일 생성
words.txt 추가해 줌