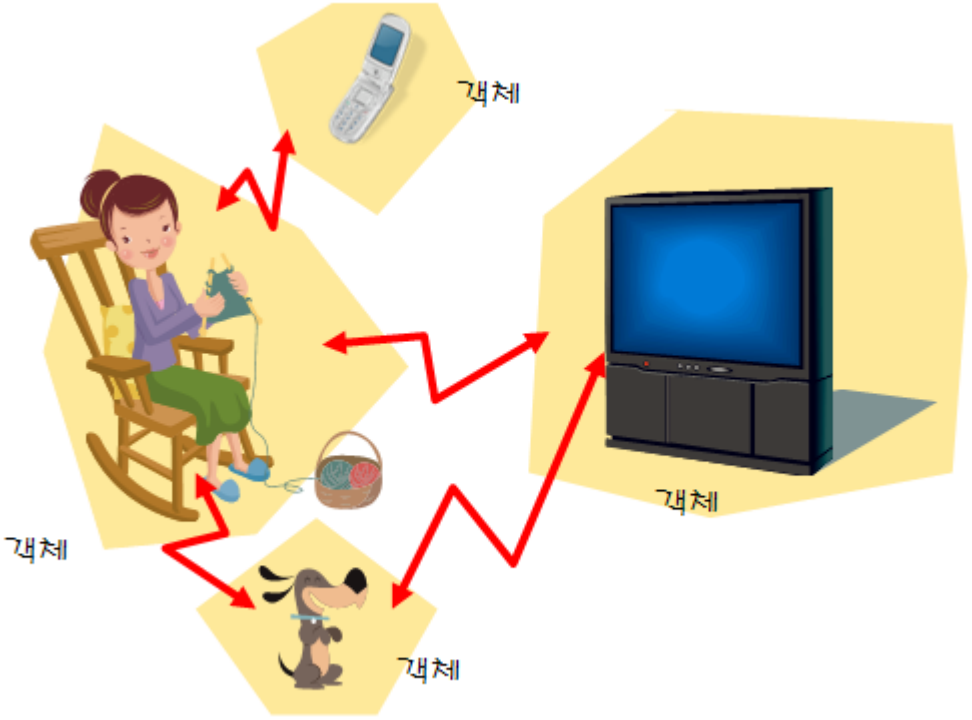
C++ Espresso

제11_2장 예외 처리



이번 장에서 학습할 내용

- 다중 catch 문장
- 자신의 예외 클래스 작성

복습 :다음 코드의 출력결과를 예상해보고 각 throw문이 던지는 예외 값을 받는 catch블록을 매칭해보자

```
try{
    throw 3;
    try{
        throw "abc";
        throw 5;
    }
    catch (int inner){
        cout << inner << endl;
    }
}
catch (char* s){
    cout << s;
}
catch (int outer){
    cout << outer << endl;
}
```

try 블록에 연결된 catch 블록으로 점프

바깥 try 블록에 연결된 catch 블록으로 점프

실습

복습 : 문자열을 정수로 변환하는 프로그램을 작성해보자. (정수로 변환할 수 없는 문자열의 경우 예외 처리하라)

```
int stringToInt(char x[]) {
    int sum = 0, len = strlen(x);
    for (int i = 0; i<len; i++) {
        if (x[i] >= '0' && x[i] <= '9')
            sum =
        else
            throw x; // char* 타입의 예외 발생
    }
    return sum;
}
```

```
int main() {
    int n;
    try {
        n = stringToInt("123"); // 문자열을 정수로 변환하는 함수
        cout << "\"123\" 은 정수 " << n << "로 변환됨" << endl;
        n = stringToInt("1A3");
        cout << "\"1A3\" 은 정수 " << n << "로 변환됨" << endl;
    }
    catch (        ) {
        cout << s << "처리에서 예외 발생!!" << endl;
        return 0;
    }
}
```

```
"123" 은 정수 123로 변환됨
1A3처리에서 예외 발생!!
계속하려면 아무 키나 누르십시오 . . .
```

예외를 발생시키는 함수의 선언

- 예외를 발생시키는 함수는 다음과 같이 선언 가능
  - 함수 원형에 연이어 throw(예외 타입, 예외 타입, ...) 선언

```
int max(int x, int y) throw(int) {
  if(x < 0) throw x;
  else if(y < 0) throw y;
  else if(x > y) return x;
  else return y;
}
```

모두 int 타입 예외 발생

```
double valueAt(double *p, int index) throw(int, char*) {
  if(index < 0)
    throw "index out of bounds exception";
  else if(p == NULL)
    throw 0;
  else
    return p[index];
}
```

char* 타입 예외 발생

int 타입 예외 발생

  - 장점
    - 프로그램의 작동을 명확히 함
    - 프로그램의 가독성 높임

다중 catch 문장

- **하나의 try 블록**에서는 **여러 개의 throw 문장**을 가질 수 있다.
- 여러 가지 타입의 값을 처리하려면 여러 개의 catch 블록을 두어야 한다.
- 예를 들어서 피자 나누기 예제에서 사람 수가 0이 될 수도 있고 사람 수가 음수가 될 수도 있다. 이것을 구분하여서 처리하려면 다음과 같이 두 개의 catch 블록을 정의하여야 한다.

다중 catch 문장

- 다음 throw 문이 던지는 예외 값을 받는 catch 블록은?

```
int main()
{
    int pizzaSlices = 12;
    int persons = 0;
    int slicesPerPerson = 0;

    try{
        cout << "사람수를 입력하시오:";
        cin >> persons;

        if (persons < 0) throw "음수";
        if (persons == 0) throw persons;

        slicesPerPerson = pizzaSlices / persons;
        cout << "한사람당 피자는 " << slicesPerPerson << "입니다." << endl;
    }
    catch (char *e){
        cout << "오류!! 사람수가 " << e << "입니다." << endl;
    }
    catch (int e){
        cout << "오류!! 사람이" << e << "명입니다." << endl;
    }
    return 0;
}
```

```
사람수를 입력하시오:-5
오류!! 사람수가 음수입니다.
계속하려면 아무 키나 누르십시오 . . .
```

```
사람수를 입력하시오:0
오류!! 사람이0명입니다.
계속하려면 아무 키나 누르십시오 . . .
```

throw 사용 시 주의 사항

- throw 문의 위치
  - 항상 try { } 블록 안에서 실행되어야 함
    - 시스템이 abort() 호출, 강제 종료
- 예외를 처리할 catch()가 없으면 프로그램 강제 종료
- catch() { } 블록 내에도 try { } catch() { } 블록 선언 가능

```
throw 3; // 프로그램이 비정상 종료된다.
try {
                ...
}
catch(int n) {
                ...
}
```

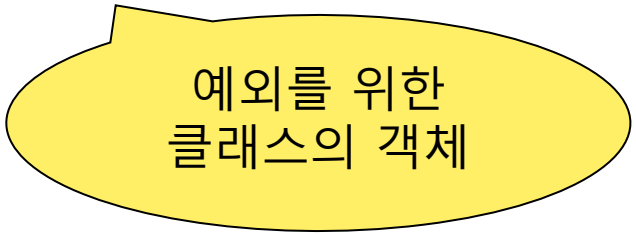
```
try {
    throw "aa"; // char* 타입의 예외를 처리할
                // catch() { } 블록이 없기 때문에
                // 프로그램 종료
}
catch(double p) {
    ...
}
```

```
try {
    throw 3;
}
catch(int x) {
    try {
        throw "aa"; // 아래의 catch(char* p) { }
                    //  블록에서 처리된다.
    }
    catch(char* p) {
        ...
    }
}
```

자신의 예외 클래스 작성

- throw 문장은 클래스 타입의 객체도 던질 수 있다.
- 예외 클래스도 단지 하나의 클래스에 불과

(예) throw NoPersonException(persons);



자신의 예외 클래스 만들기

- 예외 값의 종류
  - 기본 타입의 예외 값
    - 정수, 실수, 문자열 등 비교적 간단한 예외 정보 전달
  - 객체 예외 값
    - 예외 값으로 객체를 던질 수 있다.
    - 예외 값으로 사용할 예외 클래스 작성 필요
- 예외 클래스
  - 사용자는 자신 만의 예외 정보를 포함하는 클래스 작성
  - throw로 객체를 던짐
    - 객체가 복사되어 예외 파라미터에 전달

예외 클래스 예제

```
class NoPersonException {
    int persons;
public:
    NoPersonException(int p) { persons = p; }
    int getPersons() { return persons; }
};
```

```
int main()
{
    int pizzaSlices = 12;
    int persons = 0;
    int slicesPerPerson = 0;
    try {
        cout << "사람수를 입력하시오: ";
        cin >> persons;
        if (persons <= 0) throw NoPersonException(persons);
        slicesPerPerson = pizzaSlices / persons;
        cout << "한사람당 피자는 " << slicesPerPerson << "입니다." << endl;
    }
    catch (NoPersonException e) {
        cout << "오류: 사람이 " << e.getPersons() << "명 입니다." << endl;
    }
    return 0;
}
```

```
사람수를 입력하시오: 0
오류: 사람이 0명 입니다.
계속하려면 아무 키나 누르십시오 . . .
```

상속 관계에 있는 예외 클래스

- 출력 결과는?

```
class ParentException {
public:
    void display()
    {
        cout << "ParentException" << endl;
    }
};
class ChildException : public ParentException {
public:
    void display()
    {
        cout << "ChildException" << endl;
    }
};
```

```
int main()
{
    try {
        throw ChildException();
    }
    catch(ParentException e){
        e.display();
    }
    catch (ChildException e) {
        e.display();
    }

    return 0;
}
```

```
ParentException
계속하려면 아무 키나 누르십시오 . . .
```

상속 관계에 있는 예외 클래스

- 출력 결과는?

```
class ParentException {
public:
    void display()
    {
        cout << "ParentException" << endl;
    }
};
class ChildException : public ParentException {
public:
    void display()
    {
        cout << "ChildException" << endl;
    }
};
```

```
int main()
{
    try {
        throw ChildException();
    }
    catch (ChildException e) {
        e.display();
    }
    catch(ParentException e){
        e.display();
    }

    return 0;
}
```

```
ChildException
계속하려면 아무 키나 누르십시오 . . .
```

구체적인 예외를 먼저 잡는다.

```
try {
   getInput();
}
catch(TooSmallException e) {
   //TooSmallException만 잡힌다.
}
catch(...) {
   //TooSmallException을 제외한 나머지 예외들이 잡힌다.
}
```

반대로 하면

```
try {
   getInput();
}
catch(...) {
   //모든 예외들이 잡힌다.
}
catch(TooSmallException e) {
   //아무 것도 잡히지 않는다!
}
```

예외 클래스 예제

실습

두 양수를 입력 받아 나누기한 결과를 출력하는 프로그램에 MyException을 상속받는 다음 두개의 클래스를 작성해보자.

- 음수가 입력된 경우처리하는 클래스
  - 클래스명 : InvalidInputException
  - 매개변수 없는 생성자에서 "음수 입력 예외 발생"출력
- 0으로 나누기가 발생하는 경우를 처리하는 클래스
  - 클래스명 : DivideByZeroException
  - 매개변수 없는 생성자에서 "0으로 나누는 예외 발생"출력

```
class MyException {
    string msg;
public:
    MyException() { }
};
```

```
int main() {
    int x, y;
    try {
        cout << "두 개의 양의 정수를 입력하세요>>";
        cin >> x >> y;
        if (x < 0 || y < 0)
            throw InvalidInputException();
        if (y == 0)
            throw DivideByZeroException();

        cout << (double)x / (double)y;
    }
    catch (DivideByZeroException &e) { }
    catch (InvalidInputException &e) { }
    return 0;
}
```

```
두 개의 양의 정수를 입력하세요>>15 5
3계속하려면 아무 키나 누르십시오 . . .
```

```
두 개의 양의 정수를 입력하세요>>15 0
0으로 나누는 예외 발생
계속하려면 아무 키나 누르십시오 . . .
```

```
두 개의 양의 정수를 입력하세요>>15 -5
음수 입력 예외 발생
계속하려면 아무 키나 누르십시오 . . .
```

정답

```
class DivideByZeroException : public MyException {
public:
    DivideByZeroException() { cout << "0으로 나누는 예외 발생" << endl; }
};

class InvalidInputException : public MyException {
public:
    InvalidInputException() { cout << "음수 입력 예외 발생" << endl; }
};
```

Q & A