CPP Module 05

bir "catch" bloğundan sonra diğer "catch" bloğları otomatik olarak yürütülmez. Eğer bir hata fırlatıldı ve bir "catch" bloğu tarafından yakalandı, o hata için tanımlanmış olan "catch" bloğu çalışır ve diğer "catch" bloğları işlenmez.

Bu nedenle, hata fırlatılan türüne göre en uygun "catch" bloğunu belirlemek önemlidir. Eğer birden fazla "catch" bloğu mevcutsa ve fırlatılan hata türüne uygun bir "catch" bloğu bulunamazsa, program hata verir.

Eğer `value` değişkeni `const` olsaydı, o zaman sınıfın içinde de değerin değiştirilmesine izin verilmez. Bu nedenle, set fonksiyonunun kullanılması yasak olacaktır. Bunun yerine, sınıfın constructor fonksiyonlarından birini kullanarak değerin değiştirilmesi gerekir.

csharp Copy code

```
class Example {
private:
  const int value;

public:
  Example(int v) : value(v) {}

  int getValue() const {
    return value;
  }
};
```

Bu örnekte, `value`
değişkeni `const`
olarak tanımlanmıştır ve sadece constructor fonksiyonları tarafından başlangıç değeri verilebilir. `getValue`
fonksiyonu hala kullanılabilir, ancak değer değiştirilemez.

```
#include <iostream>
#include <exception>
#include <string>

class CustomException : public std::exception
{
    private:
        std::string message;
    public:
        CustomException(const std::string & msg) : message(msg) {}
        const char * what() const throw() { return message.c_str(); }
};

class MyClass
{
    private:
        int value;
    public:
        MyClass(int val)
        {
            if (val < 0)
                throw CustomException("Value cannot be negative");
            value = val;
        }
};
```

```
try
{
    MyClass myObject(-1);
}
catch (const CustomException & e)
{
    std::cout << e.what() << std::endl; // Output: Value cannot be negative
}
```

https://en.wikipedia.org/wiki/Halting_problem

To initialize the const value using constructor, we have to use the initialize list. This initializer list is used to initialize the data member of a class. The list of members, that will be initialized, will be present after the constructor after colon. members will be separated using comma.

```
#include <iostream>
using namespace std;
class MyClass{
   private:
      const int x;
   public:
      MyClass(int a) : x(a){
      //constructor
   }
};
```

In both examples, the `const_value_`
 member variable is assigned a value based on the `value`
 argument passed to the constructor. If `value`
 is greater than 0, `const_value_`
 is set to `value`
. Otherwise, it is set to 0.

```cpp
class MyClass {
 public:
  MyClass(int value) {
    if (value > 0) {
      const_value_ = value;
    } else {
      const_value_ = 0;
    }
  }

 private:
  const int const_value_;
};
```

```cpp
class MyClass {
 public:
  MyClass(int value) : const_value_(value > 0 ? value : 0) {}

 private:
  const int const_value_;
};
```

```cpp
Form::Form(const std::string sName, int sReqGradeSign, int sReqGradeExe) : name(sName),
reqGradeSign(sReqGradeSign > 150 ? 150 : (sReqGradeSign < 1) ? 1 : sReqGradeSign),
reqGradeExe(sReqGradeExe > 150 ? 150 : (sReqGradeExe < 1) ? 1 : sReqGradeExe)
{
    //this->setName(sName);
    try
    {
        if (sReqGradeSign < 1 || sReqGradeExe < 1)
        {
            throw GradeTooHighException();
        }
        if (sReqGradeSign > 150 || sReqGradeExe > 150)
        {
            throw GradeTooLowException();
        }
    }
    catch(const std::exception& e)
    {
        std::cerr << e.what() << '\n';
    }
    this->setIsSigned(false);
    std::cout << "\033[1;34mForm CONSTRUCTOR CALLED\033[0m" << std::endl;
}
```