



necatiergin / cpp_kursu_odevleri

Sign up



Code

Issues 0

Pull requests 0

Projects 0

Actions

Wiki

Security

Pulse

Branch: master

Find file

Copy path

cpp_kursu_odevleri / templates / templates_09.md

Fetching contributors...



62 lines (48 sloc) | 1.51 KB

Raw

Blame

History



Bu çalışma sorusu *Herb Sutter*'ın "*More exceptional C++*" isimli kitabından alınarak değiştirilmiştir. Aşağıdaki kodun standart çıkış akımına ne yazdıracağını tahmin etmeye çalışın. Daha sonra kodu derleyip çalıştırın. Yanıtınız %100 doğru ise C++ bilginiz ile gurur duyabilirsiniz. Her bir işlev çağrısı için derleyicinizin işlev şablon yüklemeleri ve özelleştirmeleri (*function template overloading* – *function template specialization*) arasından yaptığı seçimin nedenini anlamaya ve açıklamaya çalışın.

```

#include <cstdio>
#include <complex>

template<typename T1, typename T2> //0
void f(T1, T2) { putchar('0');}

template<typename T> //1
void f(T) { putchar('1'); }

template<typename T> //2
void f(T, T){ putchar('2'); }

template<typename T> //3
void f(T *) { putchar('3'); }

template<typename T> //4
void f(T*, T) { putchar('4'); }

template<typename T> //5
void f(T, T*) { putchar('5'); }

template<typename T> //6
void f(int, T *) { putchar('6'); }

template<> //7
void f<int>(int) { putchar('7'); }

void f(int, double) //8
{
    putchar('8');
}

void f(int) //9
{
    putchar('9');
}

int main()
{
    int          i = 10;
    double       d = 2.3;
    float        ff = 7.8f;
    std::complex<double> c;

    f(i);        // a
    f<int>(i);    // b
    f(i, i);     // c
    f(c);        // d
    f(i, ff);    // e
    f(i, d);     // f
    f(c, &c);    // g
    f(i, &d);    // h
    f(&d, d);    // i
    f(&d);       // j
    f(d, &i);    // k
    f(&i, &i);    // l
}

```

- [Terms](#)
- [Privacy](#)
- [Security](#)
- [Status](#)
- [Help](#)
- [Contact GitHub](#)
- [Pricing](#)
- [API](#)
- [Training](#)
- [Blog](#)
- [About](#)