深蓝学院
shenlanxueyuan.com

Project5 思路讲解

主讲人 苏涛

纲要

深蓝学院
shenlanxueyuan.com

- 第一部分：题目要求
- 第二部分：长整数加法
- 第三部分：进制转换

题目要求

- 编译期实现两个10进制非负整数的相加
- 扩展题：进制转换

纲要

- 第一部分：题目分析
- **第二部分：长整数加法**
- 第三部分：进制转换

长整数加法

1、整体步骤：

获取整数1；
获取整数2；
翻转整数1；
翻转整数2；
翻转后的整数累加；
结果翻转；
打印；

需要完成**数组定义、翻转、累加、打印**这些功能。

长整数加法

2、数组的定义（变长模板）

```
template <unsigned...> struct Cont;
```

3、数组翻转

```
template <typename Res, typename Remain>
struct Reverse {
    using type = Res;
};
template <unsigned... Processed, unsigned T1, unsigned... Remain>
struct Reverse<Cont<Processed...>, Cont<T1, Remain...>> {
    using type = ...;
};

template <typename T>
using reverse = typename Reverse<Cont<>, T>::type;
```

长整数加法

4、数组累加（列举所有的情况）

数组一是否有值、数组二是否有值，组合就是四种情况

```
template <typename Res, typename Remain1, typename Remain2, unsigned C>
struct Add;
// 数组一和数组二都有值
template <unsigned... Processed, unsigned Num1, unsigned... Remain1, unsigned Num2,
          unsigned... Remain2, unsigned C>
struct Add<Cont<Processed...>, Cont<Num1, Remain1...>, Cont<Num2, Remain2...>, C> {
    using type = std::conditional_t...
};
// 数组一没值，数组二有值
template <unsigned... Processed, unsigned Num2, unsigned... Remain2, unsigned C>
struct Add<Cont<Processed...>, Cont<>, Cont<Num2, Remain2...>, C> {
};
```

长整数加法

5、数组打印（可以偏特化、折叠表达式）

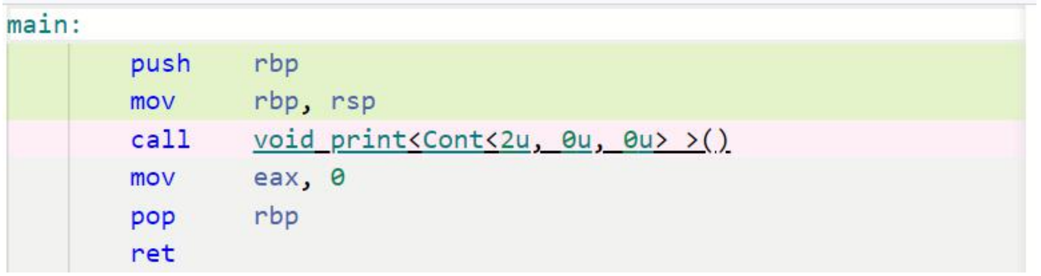
```
template <typename T>
struct Print;
template <unsigned... x>
struct Print<Cont<x...>> {
    static void print() {
        ((std::cout << x << ' '),...) << std::endl;
    }
};

template <typename T>
void print() {
    Print<T>::print();
}
```

长整数加法

6、调用逻辑

```
using Input1 = Cont<1, 9, 9>;
using Input2 = Cont<1>;
using Rev1 = reverse<Input1>;
using Rev2 = reverse<Input2>;
using RevRes = add<Rev1, Rev2>;
using Res = reverse<RevRes>;
print<Res>();
```



```
main:
        push    rbp
        mov     rbp, rsp
        call    void print<Cont<2u, 0u, 0u> >()
        mov     eax, 0
        pop     rbp
        ret
```

纲要

- 第一部分：题目分析
- 第三部分：思路讲解
- 第三部分：进制转换

进制转换

声明转换类模板：

```
template <typename Res, typename Dividend, typename Quotient, unsigned M,
          unsigned N, unsigned Remainder>
struct Convert;
```

需要完成：

1、递归进行除法操作，直到被除数Dividend没值为止、最终获得商Quotient和余数Remainder

2、去除商的前导0

3、如果商还有值，将商作为新的被除数，将余数逆序添加到结果中

4、商没有值，直接获取到结果

进制转换

1、递归进行除法操作

```
template <unsigned... Processed, unsigned T, unsigned... Dividend, unsigned... Quotient,
          unsigned Remainder, unsigned M, unsigned N>
struct Convert<Cont<Processed...>, Cont<T, Dividend...>, Cont<Quotient...>, Remainder, M, N> {};
```

2、去除商的前导0

```
template <unsigned... Processed, unsigned... Quotient, unsigned Remainder, unsigned M, unsigned N>
struct Convert<Cont<Processed...>, Cont<>, Cont<0, Quotient...>, Remainder, M, N> {};
```

3、商还有值

```
template <unsigned... Processed, unsigned T, unsigned... Quotient, unsigned Remainder, unsigned M,
         unsigned N>
struct Convert<Cont<Processed...>, Cont<>, Cont<T, Quotient...>, Remainder, M, N> {};
```

4、商没有值

```
template <unsigned... Processed, unsigned Remainder, unsigned M, unsigned N>
struct Convert<Cont<Processed...>, Cont<>, Cont<>, Remainder, M, N> {};
```

深蓝学院
shenlanxueyuan.com

感谢各位聆听！
Thanks for Listening