PrincipleOfCompilingFinal

2021-2022学年第二学期

实验报告

- 课程名称:编程语言原理与编译
- 实验项目: 编译原理大作业：Micro C初实现
- 实验指导教师:张芸
-

姓名	学号	班级
夏鹏程	31901117	计算1904
吴鸿飞	31901085	计算1903

- 项目源代码:https://gitee.com/Shylock-patriot/microc

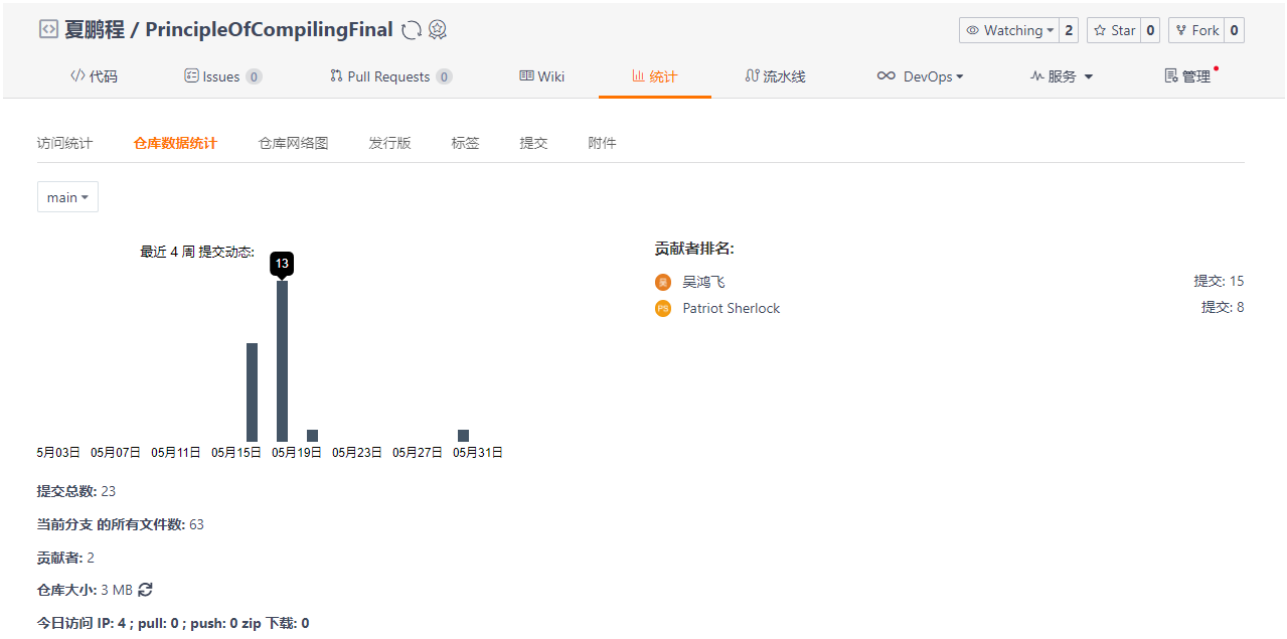
一、项目评价基本情况

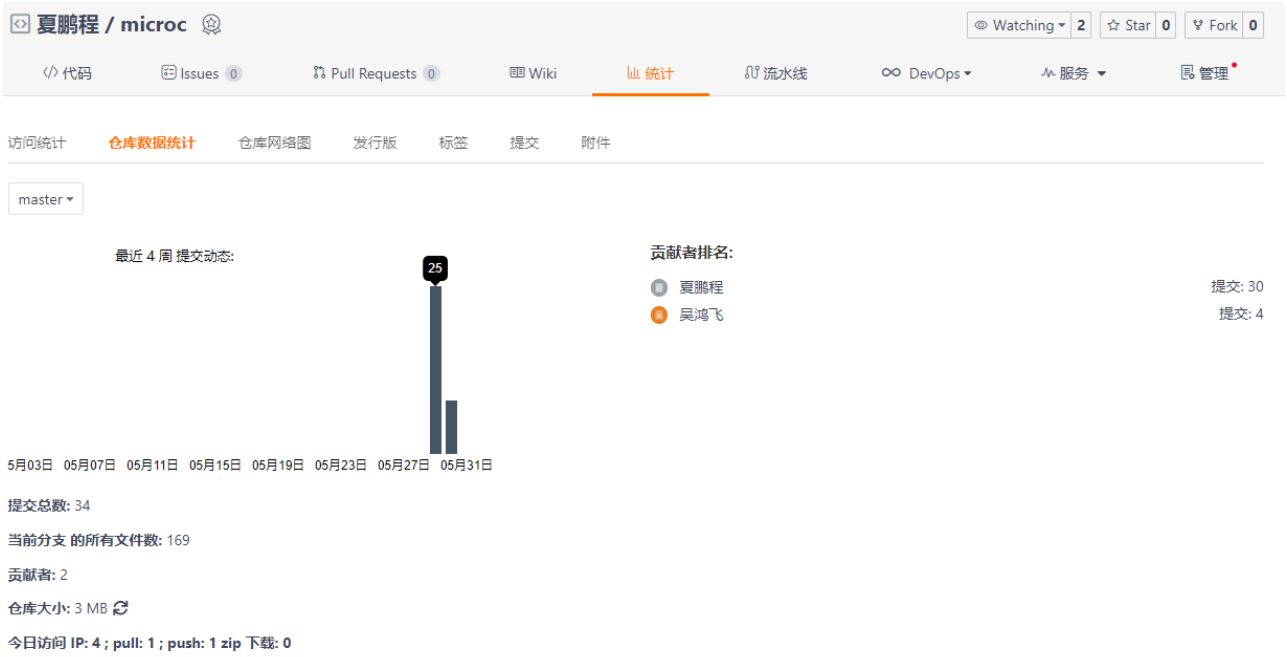
（一）成员贡献情况

姓名	任务	权重
夏鹏程	编译器及其测试	0.95
吴鸿飞	解释器及其测试	0.95

（二）成员提交代码日志

先前仓库存在文件不足问题，原仓库为"Shylock-patriot/PrincipleOfCompilingFinal"，后将其迁移到"Shylock-patriot/microc"。





（三）项目自评等级

1.解释器

词法	评分	备注
注释 //,/**/	★★★	

词法	评分	备注
语法		
for 循环	★★★★	
while循环	★★★	
dowhile循环	★★★★★	
语义		
i++,i--,	★★★	
++i,--i	★★★	
+= -= *= /= %=	★★★	
三元运算符 ?:	★★★★★	

2.编译器

词法语法	评分	备注
break	★★★★	
continue	★★★	
float类型	★★★★	
i++, ++i, i--, --i	★★★	
+= -= *= /= %=	★★★	
"左移"运算	★★	
"或"运算	★★	
"异或"运算	★★	
"取反"运算	★★	

二、项目运行命令

> 以*ex1.c*为例

1.解释器

```
dotnet clean  interpc.fsproj      # 可选
dotnet build -v n interpc.fsproj # 构
建./bin/Debug/net6.0/interpc.exe ，-v n查看详细生成过程
./bin/Debug/net6.0/interpc.exe example/ex1.c 8 # 执行解释器
```

2.编译器

```
dotnet clean  microc.fsproj      # 可选
dotnet build  microc.fsproj    # 构建 ./bin/Debug/net6.0/microc.exe
dotnet run --project microc.fsproj example/ex1.c     # 执行编译器，编译
ex1.c，并输出ex1.out 文件
javac Machine.java
java Machine ./example/ex1.out 3
```

三、测试

（一）解释器

1.注释

```
// micro-C example 1
int g ;
int h[3] ;
void main(int n) {
  h[0] = 1;
  // h[4] = 5;
  // print h[3]; //数组首地址
  // print h[4]; //参数 n
  // h[5] = 5;
  // h[6] = 5;
  // h[7] = 5;
  // h[9] = 5;
  // h[10] = 5;
  // h[11] = 5;
  // h[12] = 5;
  // 数组越界，程序的行为会异常
  // 此时解释器，栈式虚拟机，x86程序表现各不相同，可以思考原因

  while (n > 0) {
```

```
    print n;
    n = n - 1;
  }
  println;
}
```

运行结果

```
PS D:\repository\project\Git\PrincipleOfCompilingFinal> dotnet run --project interpc.fsproj example/ex1.c 8
Micro-C interpreter v 1.1.0 of 2021-5-19
interpreting example/ex1.c ...inputargs:[8]
8 7 6 5 4 3 2 1
PS D:\repository\project\Git\PrincipleOfCompilingFinal>
```

2.while

```
void main(int n ) {
    while(n){
     print n;
     n--;
    }
}
```

运行结果

```
PS D:\repository\project\Git\PrincipleOfCompilingFinal> dotnet run --project interpc.fsproj example/while.c 8
Micro-C interpreter v 1.1.0 of 2021-5-19
interpreting example/while.c ...inputargs:[8]
8 7 6 5 4 3 2 1
PS D:\repository\project\Git\PrincipleOfCompilingFinal>
```

3.for

```
void main(int n) {
  int i;
  for(i=0;i<n;i++){
    print i;
  }
}
```

运行结果

```
PS D:\repository\project\Git\PrincipleOfCompilingFinal> dotnet run --project interpc.fsproj example/for.c 8
Micro-C interpreter v 1.1.0 of 2021-5-19
interpreting example/for.c ...inputargs:[8]
1 2 3 4 5 6 7
PS D:\repository\project\Git\PrincipleOfCompilingFinal>
```

4.dowhile

```
void main(int n ) {
    do{
     print n;
     n--;
    }while(n)
}
```

运行结果

```
PS D:\repository\project\Git\PrincipleOfCompilingFinal> dotnet run --project interpc.fsproj example/dowhile.c 8
Micro-C interpreter v 1.1.0 of 2021-5-19
interpreting example/dowhile.c ...inputargs:[8]
8 7 6 5 4 3 2 1
PS D:\repository\project\Git\PrincipleOfCompilingFinal>
```

5.+= -= *= /= %=

```
void main(int n) {
  n=10;
  print n;
  n+=3;
  print n;
  n-=4;
  print n;
  n*=3;
  print n;
  n/=2;
  print n;
  n%=2;
  print n;
}
```

运行结果

```
PS D:\repository\project\Git\PrincipleOfCompilingFinal> dotnet run --project interpc.fsproj example/_+=.c 8
Micro-C interpreter v 1.1.0 of 2021-5-19
interpreting example/_+=.c ...inputargs:[8]
10 13 9 27 13 1
PS D:\repository\project\Git\PrincipleOfCompilingFinal>
```

6.i++ i-- ++i --i

```
void main(int n) {
  n++;
  ++n;
  print n;
  n=10;
  n--;
  --n;
  print n;
}
```

运行结果

```
PS D:\repository\project\Git\PrincipleOfCompilingFinal> dotnet run --project interpc.fsproj example/nPP.c 8
Micro-C interpreter v 1.1.0 of 2021-5-19
interpreting example/nPP.c ...inputargs:[8]
10 8
PS D:\repository\project\Git\PrincipleOfCompilingFinal>
```

7.三元运算符

```
void main(int n) {
  int a;
  a=n?101:110;
  print a;
  println;
}
```

运行结果

```
PS D:\repository\project\Git\PrincipleOfCompilingFinal> dotnet run --project interpc.fsproj example/3operator.c 8
Micro-C interpreter v 1.1.0 of 2021-5-19
interpreting example/3operator.c ...inputargs:[8]
101
PS D:\repository\project\Git\PrincipleOfCompilingFinal>
```

（二）编译器

1.break continue

（1）break

(1.1)测试代码

```
void main(int n) {
  int i;
  i=0;
  while (i < 3) {
    if (i == 1) {
        i = i + 2;
        print i;
         break;
    }
    print i;
    i=i+1;
  }
}
```

(1.2)运行

```
  正在确定要还原的项目…
  所有项目均是最新的，无法还原。
D:\VS-code\f#\myfork\microc\Comp.fs(180,11): warning FS0025: 此表达式中的模式匹配不完整。 例如，值“DoWhi
le (_, _)”可以指示模式未涵盖的用例。 [D:\VS-code\f#\myfork\microc\microc.fsproj]
D:\VS-code\f#\myfork\microc\Comp.fs(240,11): warning FS0025: 此表达式中的模式匹配不完整。 例如，值“Prim3
 (_, _, _)”可以指示模式未涵盖的用例。 [D:\VS-code\f#\myfork\microc\microc.fsproj]
  microc -> D:\VS-code\f#\myfork\microc\bin\Debug\net6.0\microc.dll

Compiling ./example/break_test.c ......

VM instructions saved in file:
        ./example/break_test.ins
VM instructions saved in file:
        ./example/break_test.insx86
x86 assembly saved in file:
        ./example/break_test.asm
Numeric code saved in file:
        ./example/break_test.out
Please run with VM.

sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ machine.exe ./example/break_test.out 3
bash: machine.exe: command not found

sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ machine.exe ./example/break_test.out 3
bash: machine.exe: command not found

sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ .\machine.exe ./example/break_test.out 3
bash: .machine.exe: command not found

sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ ./machine.exe ./example/break_test.out 3
0 3
```

（2）continue

(2.1)测试代码

```
void main(int n) {
  int i;
  i=0;
  while (i < 3) {
    if (i == 1) {
        i = i + 1;
        continue;
    }
    print i;
    i=i+1;
  }
}
```

(2.2)运行

```
          consider setting precedences explicitly using %left %right
etting explicit precedence on rules using %prec
          183 states
          22 nonterminals
          55 terminals
          88 productions
          #rows in action table: 183
D:\VS-code\f#\myfork\microc\Comp.fs(180,11): warning FS0025: 此表达式
le (_, _)”可以指示模式未涵盖的用例。 [D:\VS-code\f#\myfork\microc\mi
D:\VS-code\f#\myfork\microc\Comp.fs(246,11): warning FS0025: 此表达式
 (_, _, _)”可以指示模式未涵盖的用例。 [D:\VS-code\f#\myfork\microc\m
  microc -> D:\VS-code\f#\myfork\microc\bin\Debug\net6.0\microc.dll
Numeric code saved in file:
        ./example/continue_test.out
Please run with VM.

sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ ./machine.exe ./example/continue_test.out 3
0 2
sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$
```

2.~;<<;^;|; 四个运算

（1）取反运算

(1.1)测试代码

```
void main(int n) {
    int a;
    int b;
    int c;
    a = 1;
    b = 0;
    a = ~a;
}
```

(1.2)运行

```
sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ dotnet run -p microc.fsproj ./example/bitnot.c 3
警告 NETSDK1174: 已弃用使用缩写“-p”来代表“--project”。请使用“--project”。
Micro-C Stack VM compiler v 1.2.0 of 2021-5-12
Compiling ./example/bitnot.c ......

VM instructions saved in file:
        ./example/bitnot.ins
Numeric code saved in file:
        ./example/bitnot.out
Please run with VM.

sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ ./machine.exe ./example/bitnot.out 3
Segmentation fault
```

（**1.3**）问题

虽然在编译过程中没出现问题，生成.out文件，但输出结果不是我们需要的:0，而是提示Segmentation fault，这个问题需要修改

（**2**）**<<** 操作

(2.1)测试代码

```
void main(int n) {
    int a;
    int b;
    int c;
    a = 1;
    b = 0;
//  c = a&b;
//   print c;
//  c = a|b;
//   print c;
    c = a << 2;
    print c;
//   c = c >> 1;
//   print c;
//   c = a ^ b;
//   print c;
//   c = ~b;
//   print c;
```

```
}
```

(2.2)运行

```
sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ dotnet run -p microc.fsproj ./example/bit.c
警告 NETSDK1174: 已弃用使用缩写“-p”来代表“--project”。请使用“--proje
Micro-C Stack VM compiler v 1.2.0 of 2021-5-12
Compiling ./example/bit.c ......

VM instructions saved in file:
        ./example/bit.ins
Numeric code saved in file:
        ./example/bit.out
Please run with VM.

sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ ./machine.exe ./example/bit.out
3
sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ dotnet clean  microc.fsproj
用于 .NET 的 Microsoft (R) 生成引擎版本 17.0.0+c9eb9dd64
版权所有(C) Microsoft Corporation。保留所有权利。
```

（**3**）异或操作 ^

(3.1)测试代码

```
void main() {
    int a;
    int b;
    int c;
    a = 1;
    b = 0;
  c = a ^ b;
  print c;
}
```

(3.2)运行

运行前需要先

```
dotnet build microc.fsproj； dotnet build microcc.fsproj
```

```
D:\VS-code\f#\myfork\microc\Contcomp.fs(187,11): warning FS0025: 此表达
匹配不完整。 例如，值“Break”可以指示模式未涵盖的用例。 [D:\VS-code\f#\my
oc\microcc.fsproj]
D:\VS-code\f#\myfork\microc\Contcomp.fs(244,11): warning FS0025: 此表达
匹配不完整。 例如，值“Prim3 (_, _, _)”可以指示模式未涵盖的用例。 [D:\VS-
yfork\microc\microcc.fsproj]
    2 个警告
    0 个错误

已用时间 00:00:05.25

sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ dotnet run -p microc.fsproj ./example/bitxor.c
警告 NETSDK1174: 已弃用使用缩写“-p”来代表“--project”。请使用“--project”
9m
Micro-C backwards compiler v 1.0.0.0 of 2012-02-13
Compiling ./example/bitxor.c to ./example/bitxor.out

sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ ./machine.exe ./example/bitxor.out
1
sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$
```


（4）或运算 |

(4.1)测试代码

```
void main() {
    int a;
    int b;
    int c;
    a = 1;
    b = 0;
    c = a|b;
    print c;
}
```

(4.2)运行

```
sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ dotnet run -p microc.fsproj ./example/bitor.c
警告 NETSDK1174: 已弃用使用缩写“-p”来代表“--project”。请使用“--project”。
Micro-C Stack VM compiler v 1.2.0 of 2021-5-12
Compiling ./example/bitor.c ......

VM instructions saved in file:
        ./example/bitor.ins
Numeric code saved in file:
        ./example/bitor.out
Please run with VM.

sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ ./machine.exe ./example/bitor.out
1
sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$
```

3.Float浮点数

(1)测试代码

```
void main() {
   float f;
   f = 1.2;
   print f;
}
```

(2)运行

```
$ dotnet run -p microc.fsproj ./example/Float.c
警告 NETSDK1174: 已弃用使用缩
写“-p”来代表“--project”。请使用“--project”。
Micro-C Stack VM compiler v 1.2.0 of 2021-5-12
Compiling ./example/Float.c ......

VM instructions saved in file:
        ./example/Float.ins
Numeric code saved in file:
        ./example/Float.out
Please run with VM.

sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ ./machine.exe ./example/Float.out
1067030938
sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/m        aster)
$
```

4.+=;-=;=;/=

（1）测试代码

```
void main(int n) {
  n=10;
  print n;
  n+=3;
  print n;
  n-=4;
  print n;
  n*=3;
  print n;
  n/=2;
  print n;
  n%=2;
  print n;
}
```

（2）运行

```
sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ dotnet run --project microc.fsproj example/_+=.c 3
Micro-C Stack VM compiler v 1.2.0 of 2021-5-12
Compiling example//_+=.c ......

VM instructions saved in file:
        example//_+=.ins
Numeric code saved in file:
        example//_+=.out
Please run with VM.

sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ java Machine.java
错误: 找不到或无法加载主类 Machine.java

sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ javac Machine.java

sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ java Machine ./example/_+=.out 3
10 13 9 27 13 1
Ran 0.001 seconds

sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$
```

5.i++;i--;++i;--i

(1)测试代码

```
void main(int n) {
  n++;
  ++n;
  print n;
  n=10;
  n--;
  --n;
  print n;
}
```

(2)运行

```
sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ dotnet run --project microc.fsproj example/nPP.c 3
Micro-C Stack VM compiler v 1.2.0 of 2021-5-12
Compiling example/nPP.c ......

VM instructions saved in file:
        example/nPP.ins
Numeric code saved in file:
        example/nPP.out
Please run with VM.

sixia@LAPTOP-JTULJ771 MINGW64 /d/VS-code/f#/myfork/microc (master)
$ java Machine ./example/nPP.out 3
5 8
Ran 0.001 seconds
```

四、项目心得体会

（一）项目开发过程心得

1.夏鹏程心得体会

编译原理,说实话,在大三下来这么一门硬核的课程,真的感觉挺害怕的。心里既想着考研，也得兼顾平时的学业，而且编译原理又很晦涩，真的很难。对于大作业，说实话，刚接触到真的很懵，在网上找了一些代码，结合着《Programming Language Concepts for Software Developers》来学，但发现书里的内容和真的开始搞microc时差距很大，和同组伙伴交流后，结合网络上过去的提交记录进行学习，稍微掌握了整体流程: Clex.fsl作为词法定义,将字符变为代表字符的类，Cpar.fsy作为语法定义将类进行语句组合识别;而Absyn.fs作为抽象语法树,定义语句对象，具体到执行中，解释器主要围绕Interp.fsproj转悠，编译器围绕Comp.fsproj转悠。了解了整个流程，心里也稍稍有了些底。但到写代码时，每次commit虽然有成就感，但围绕着提交后源代码报错时，又不得不回退找bug，像在原地转圈一样。整个过程真的很考验心态。

2.吴鸿飞心得体会

编译原理，本来以为很简单，但学了之后我是一脸懵逼的。什么LL(1)语法分析方法，pda的转换，从最开始的一点都不了解，到后来做过题目搜过想换资料之后的渐渐熟悉。编译原理作为最后的几门课之一，并没有我想象中的学习到什么东西，相反的，甚至觉得无足轻重。因为工作之后无论做前端、后端、算法，都不需要这这些知识。毕业之后，极少数人，会创造一门语言，实现一个自己的编译器，或参与一个这样的项目。在这么重要的时间出现这样的一门课的目的与意义我不明白。对于microc，在已有项目的基础上，理解这个项目还是要依靠同学的解释。通过已有项目的提交记录，我慢慢理解项目是怎样运行的，并在已有项目基础上新增更多功能。实现while时，参考了for的实现，这是一件简单又困难的事，简单的是各种符号有迹可循，困难的是循环的具体实现。好在不算太复杂，最终还是理解了编译器的运行 并顺利的完成了项目。

（二）对本课程建议

编译原理很晦涩，真的很难。平时上课感觉老师不应该只是照着PPT去念东西，我们缺乏对编译/解释两个功能更直观的感受；大三阶段上这么一门硬核的课程，说实话难度真的好大，而且书本很多时候没怎么用到的感觉，除了平时作业偶尔需要翻书，但很多时候书本的作用没有凸现出来。