# 实验4 & 课后作业

助教：胡光能，尚迪
编译原理讲师：戴新宇
hugn@nlp.nju.edu.cn

# 实验4：目标代码生成

# 目录

- 目标代码生成
- 指令选择：中间表示到目标代码的映射
- 寄存器分配：操作数要位于寄存器中
- 栈管理：函数调用
- 系统调用：输出打印的例子
- 注意事项

# 目标代码生成

- 将实验三得到的中间表示翻译为汇编代码


- 最终能在MIPS32指令模拟器SPIM运行得到正确的结果
- 解决三个问题
  - 指令选择：中间表示映射到目标代码
  - 寄存器分配：操作数要位于寄存器中
  - 栈管理：函数调用

# 指令选择

- 将中间表示映射到目标代码
  - 将一条较为抽象的中间代码映射为一条或多条低级的目标代码
- 《实验4》表11
  - 标号、跳转
  - 算术运算
  - 逻辑运算
  - 存取指令
  - 调用返回

表11. 中间代码与MIPS32指令对应的一个示例。

| 中间代码 | MIPS32指令 |
|---|---|
| LABEL x: | x: |
| x := #k | li reg(x)[1], k |
| x := y | move reg(x), reg(y) |
| x := y + #k | addi reg(x), reg(y), k |
| x := y + z | add reg(x), reg(y), reg(z) |
| x := y - #k | addi reg(x), reg(y), -k |
| x := y - z | sub reg(x), reg(y), reg(z) |
| x := y * z[2] | mul reg(x), reg(y), reg(z) |
| x := y / z | div reg(y), reg(z)<br>mflo reg(x) |
| x := *y | lw reg(x), 0(reg(y)) |

# 寄存器分配

- 操作数要位于寄存器中
- 朴算分配算法
  - 运算前总是把操作数从内存取到寄存器中
  - 运算后总是把结果从寄存器存到内存
- 实验4的完成过程中，优先考虑实现的正确性，然后才是效率问题

# 栈管理：函数调用

- 将实验三的较为抽象的函数调用相关语句翻译为对应的汇编代码
- **{ARG, CALL} → {jal, jr}**
- 实验4的完成过程中，参数个数太多时的压栈处理作为额外的加分处理

# 系统调用：特殊的函数调用

```
1  li $v0, 4
2  la $a0, _prompt
3  syscall
```

进行了系统调用print_string(_prompt)

# 注意事项

- 抄袭

# 课后作业：偶数次作业选讲