

编译方法

本项目在 ghc 7.6.3 下进行开发。ghc 8.0.1 也可以编译通过。

假设 llc、clang 的存在。

编译 haskell 源码：

```
ghc urun.hs -O2
```

```
ghc ucomp.hs -O2
```

```
ghc ki.hs -O2
```

```
ghc kc.hs -O2
```

（可选的）推荐使用当前的 clang 产生 llvm-IR 模版

```
rm simpleruntime.ll && make
```

编译 uglyprint

```
./ucomp uglyprint.u
```

使用方法

urun 和 ucomp 是推荐的工具。

```
./urun filename.u
```

会执行 filename.u 中的程序。可以用

```
./urun helloworld.u
```

和

```
echo 1 2 | ./urun aplusb.u
```

来检查编译是否成功

```
./urun
```

会打开一个 repl 的终端。可以在其中使用 import 语句、def 语句等。例如

```
./urun
```

```
prelude>( + 1 2)
```

```
3
```

```
prelude>(def strCat ++)
```

```
prelude>(import* io)
```

```
prelude io>(putStrLn (strCat "hello" " world") exit)
```

```
hello world
```

```
prelude io>:q
```

ucomp 是编译用的工具。

```
./ucomp filename.u [-o filename] [-is arch]
```

会把 filename.u 中的程序编译到输出文件。不加 -o 会产生同名（去掉 .u 后缀）的输出文件。

可以加入 -is x86-64 来指定输出到 x86-64 平台。没有测试过指定其他平台会发生什么（取决于系统的工具链）。

ki 和 kc 是为了满足作业要求而存在的 urun 和 ucomp 的替代品。除自动化测试脚本外，不推荐使用它们。

uglyprint 的使用方法：

```
./uglyprint inputfile.u
```

或者

```
./urun uglyprint inputfile.u
```

例如：

```
$/urun uglyprint.u uglyprint.u > uglyprint1.u
```

```
$/urun uglyprint1.u uglyprint1.u > uglyprint2.u
```

```
$diff uglyprint1.u uglyprint2.u
```

uglyprint 的输出保持幂等