**Nyasm**

闲来无事，我准备做一个解释器，原本叫 **AssemblyConverter （汇编转化 | 我那翻译器翻的，还不知道对不对）**，从这个名字不难看出，其功能是将高级语言转化为汇编语言，但在这个时代也恐怕没几个人还研究汇编了。  
所以，它不仅改了名字，还改了功能，它的名字是这么来的：

nyasm = nya + asm

“nya”在我老家方言里意为“什么”，其实是读“ni nya”，但由于语速过快，简化成了“nya”。  
“asm”意思是汇编，算是保留了原本名字的一部分以作纪念。  
  
**过程**  
1. 程序完成对文件（\*.nya）的读取例如  函数：open.h/read()：

using io

fn main(){

      print("hello world");

}

2. 完成引用（using io）  
这会将 **io** 模块里的内容读取并整合进源文件中，**源文件里的代码会copy一份有另用**  函数：processing.h/Uprocessing()。

fn print(str NyaText){;}

fn input(str NyaText){;}

fn NyasmVersion(){

   print("version 0.01");

}

fn exit(int NyaExitCode){

    return NyaExitCode;

}

fn main(){

      print("hello world");

}

3. 接下来给代码进行分词 函数：token.h/tokenizer\_code()。  
它会将整合好了的代码变成这样  
[fn] [print] [(] [str] [NyaText] [)] [{] [;] [}] [fn] [input] [(] [str] [NyaText] [)] [{] [;] [}] [fn] ......  
  
4. 获取函数名、参数、代码体。美化后的输出 函数：processing.h/lex()  <=  这个函数名我不知取的是否恰当：

Function: print( str NyaText )  {

    ;

}

Function: input( str NyaText ) {

    ;

}

Function: NyasmVersion( )  {

    print("version 0.01");

}

Function: exit( int NyaExitCode )  {

     return NyaExitCode;

}

Function: main( )  {

    print("hello world");

}

这些实现中还有诸多逻辑BUG，代码不仅要能跑，还得对齐脑回路！

还有帮助信息、错误处理之类的，这里不赘述

现在只干了这么多，剩下的等8月9日再说

（2025/8/7）