CLOUD NATIVE Wasm DAY EUROPE

CLOUD NATIVE
Wasm DAY
EUROPE

0061 736D

The binary magic of Wasm

Hello there!

CLOUD NATIVE
Wasm DAY
EUROPE

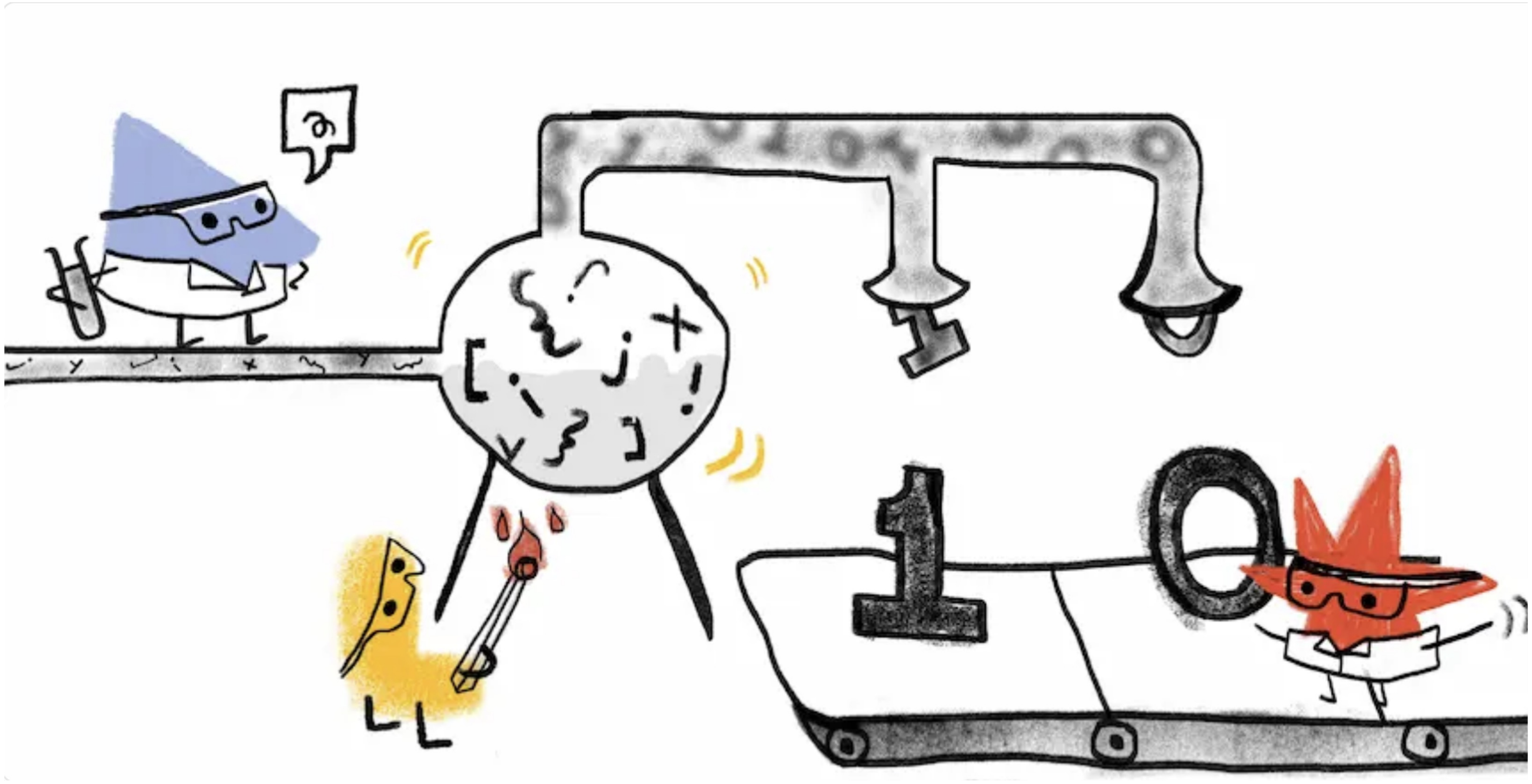
16 May 2022 I Valencia, Spain

DIVYA MOHAN
TECHNICAL WRITER
SUSE

WHO IS THIS FOR?

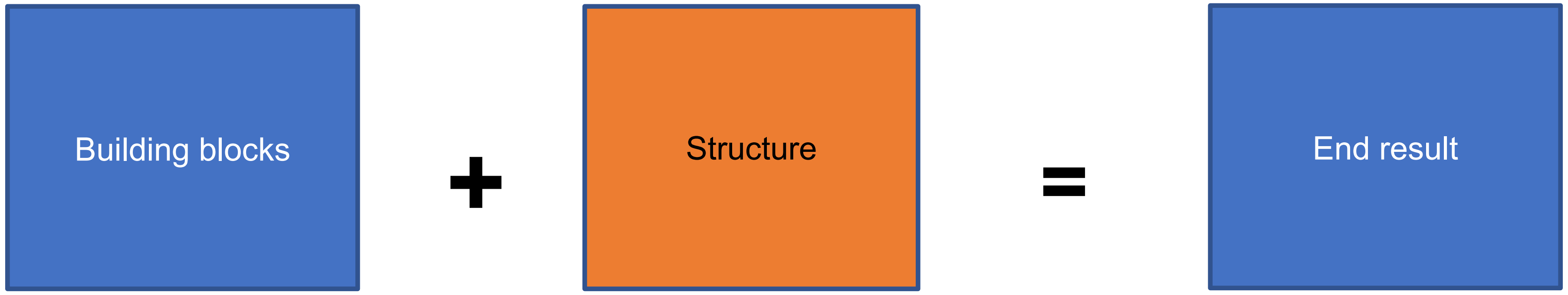
- **TL; DL: EVERYONE**
- ENTHUSIASTS/HOBBYISTS
- PEOPLE LOOKING TO:
  - WRITE WEBASSEMBLY COMPILERS
  - WRITE WEBASSEMBLY MODULES FOR OPTIMIZING COMPILER PERFORMANCE
- **POTENTIAL OVERKILL** FOR DEVELOPERS WHO WANT TO ONLY LOAD Wasm MODULES

WHY THIS? WHY NOW?



Courtesy: https://almanac.httparchive.org/en/2021/webassembly

LEARNING THE ROPES!

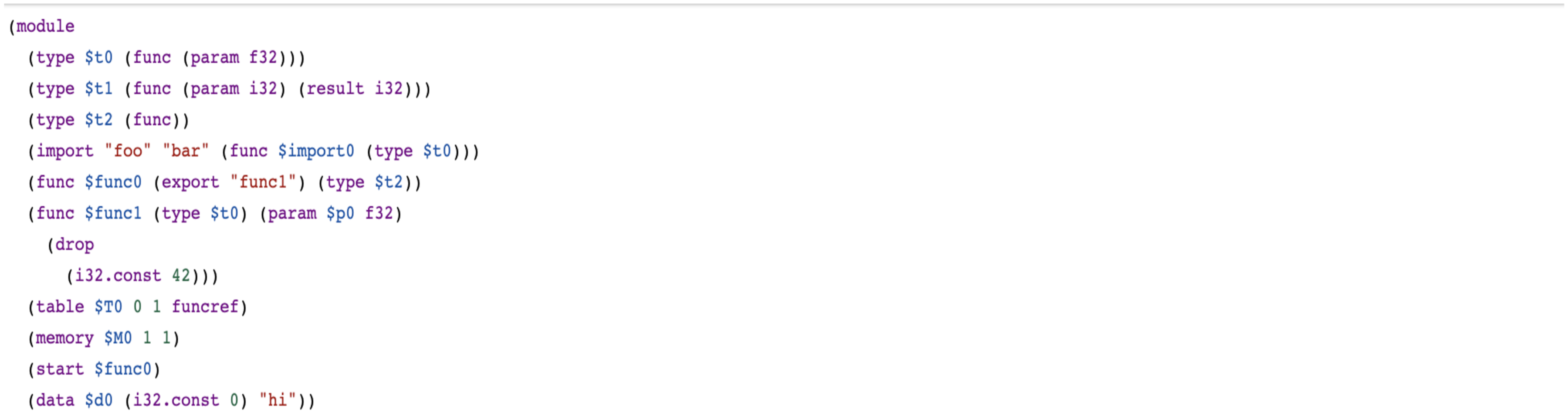


Main ingredients (i.e. variables, their types etc.)

- Grammar
- Syntax

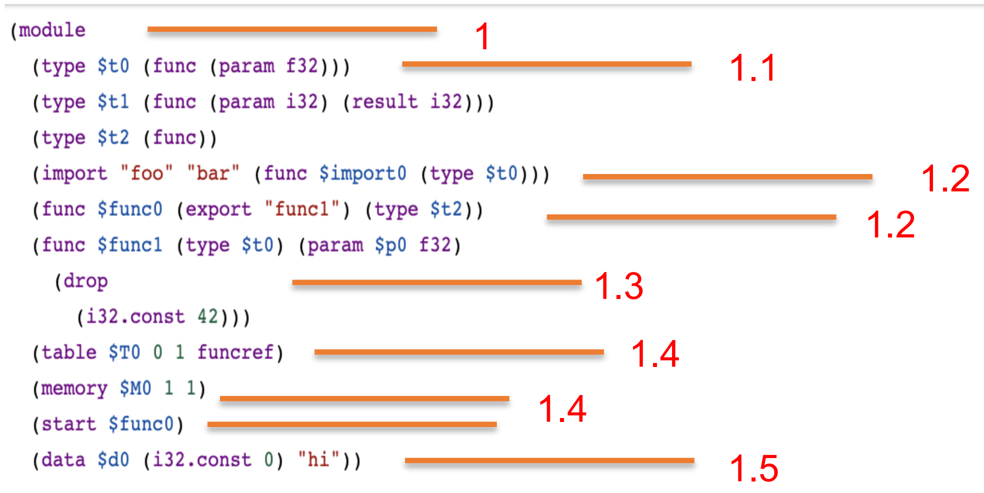
Program

A Wasm program



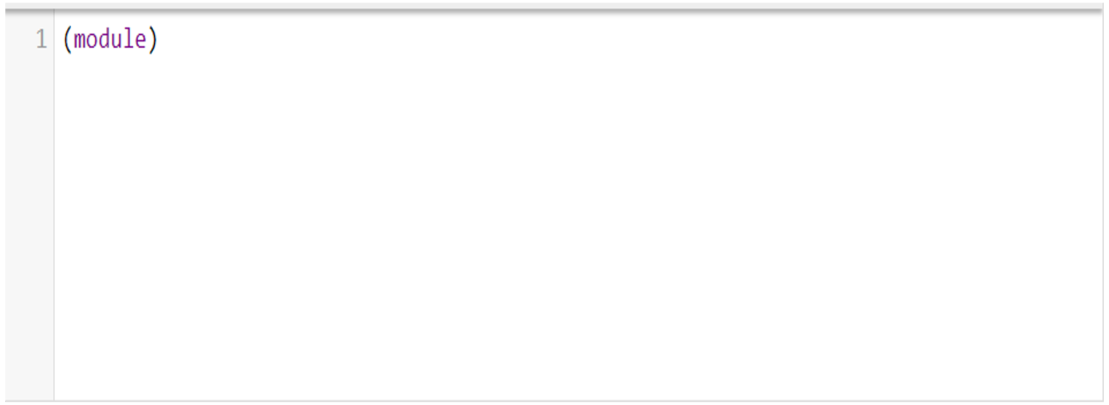
```
(module
  (type $t0 (func (param f32)))
  (type $t1 (func (param i32) (result i32)))
  (type $t2 (func))
  (import "foo" "bar" (func $import0 (type $t0)))
  (func $func0 (export "func1") (type $t2))
  (func $func1 (type $t0) (param $p0 f32)
    (drop
      (i32.const 42)))
  (table $T0 0 1 funcref)
  (memory $M0 1 1)
  (start $func0)
  (data $d0 (i32.const 0) "hi"))
```

Deconstructing a Wasm program

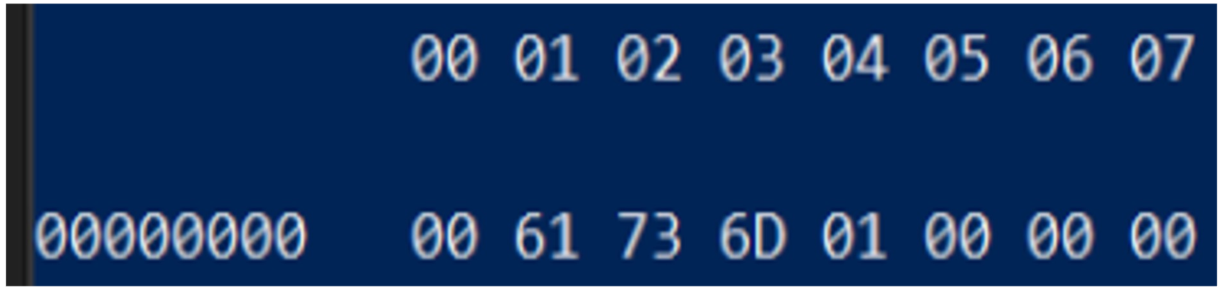


```
(module
  (type $t0 (func (param f32)))
  (type $t1 (func (param i32) (result i32)))
  (type $t2 (func))
  (import "foo" "bar" (func $import0 (type $t0)))
  (func $func0 (export "func1") (type $t2))
  (func $func1 (type $t0) (param $p0 f32)
    (drop
      (i32.const 42)))
  (table $T0 0 1 funcref)
  (memory $M0 1 1)
  (start $func0)
  (data $d0 (i32.const 0) "hi"))
```

1. MODULE



```
(module)
```

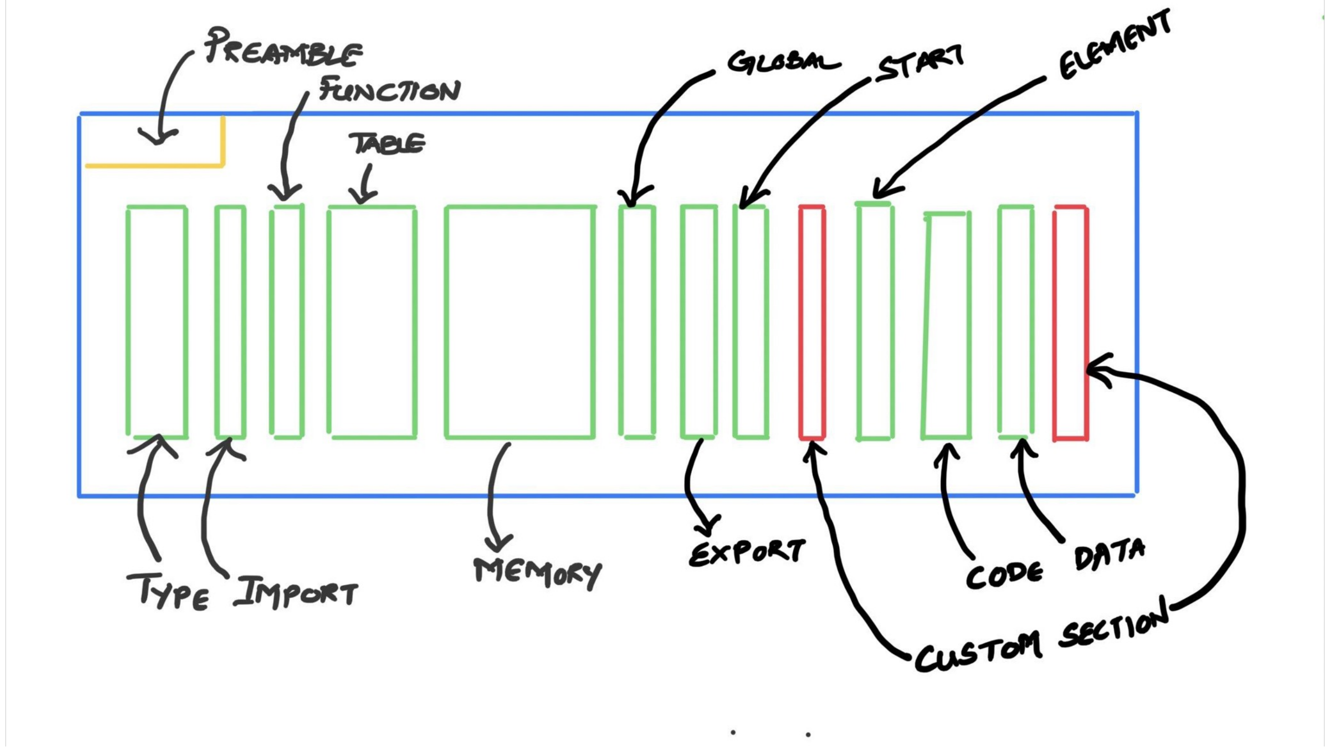


```
            00 01 02 03 04 05 06 07

00000000    00 61 73 6D 01 00 00 00
```

CLOUD NATIVE
Wasm DAY
EUROPE

Anatomy of a Wasm module



Deconstructing a Wasm program

```
(module                                                  1
  (type $t0 (func (param f32)))                          1.1
  (type $t1 (func (param i32) (result i32)))
  (type $t2 (func))
  (import "foo" "bar" (func $import0 (type $t0)))        1.2
  (func $func0 (export "func1") (type $t2))              1.2
  (func $func1 (type $t0) (param $p0 f32)
    (drop                                                1.3
      (i32.const 42)))
  (table $T0 0 1 funcref)                                1.4
  (memory $M0 1 1)
                                                         1.4
  (start $func0)
  (data $d0 (i32.const 0) "hi"))                         1.5
```

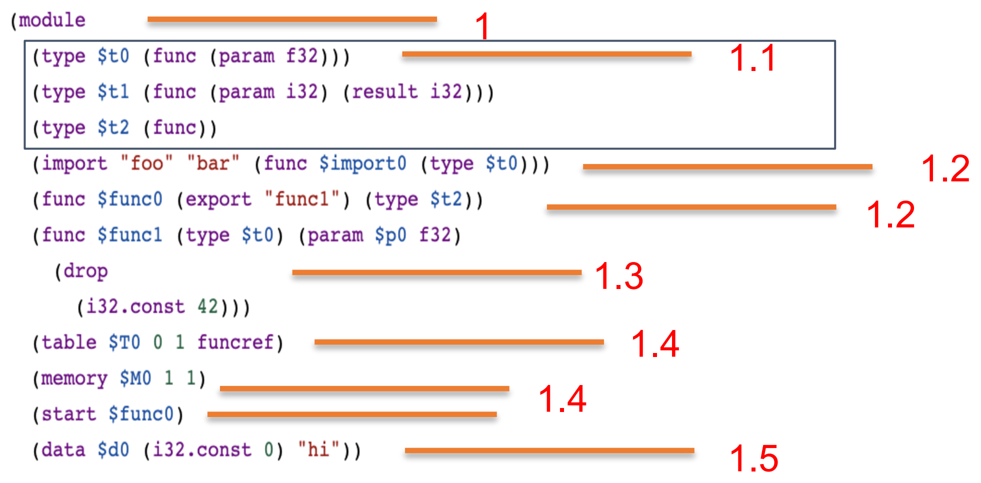
1.1 OUR OPERANDS

(A.K.A SUPPORTED TYPES)

- What are the kinds of operands in Wasm?
- How do we pass them?
- Where do they need to be defined?

Number Type	• i32|i64|f32|f64
Vector Type	• v128
Reference Type	• External/Function References
Value Type	• Can be Number/Vector/Reference Types
Result Type	• Sequence of Values i.e. resulttype ::= [vec(valtype)]
Function Type	• Signature of functions • i.e. functype ::= resulttype -> resulttype
Memory Type	• Size range i.e. memtype ::= limits
Table Type	• Reference type over a size range i.e. tabletype ::= limits reftype
Global Type	• Global variables
External Type	• Classify import & export values with respective type

Deconstructing a Wasm program



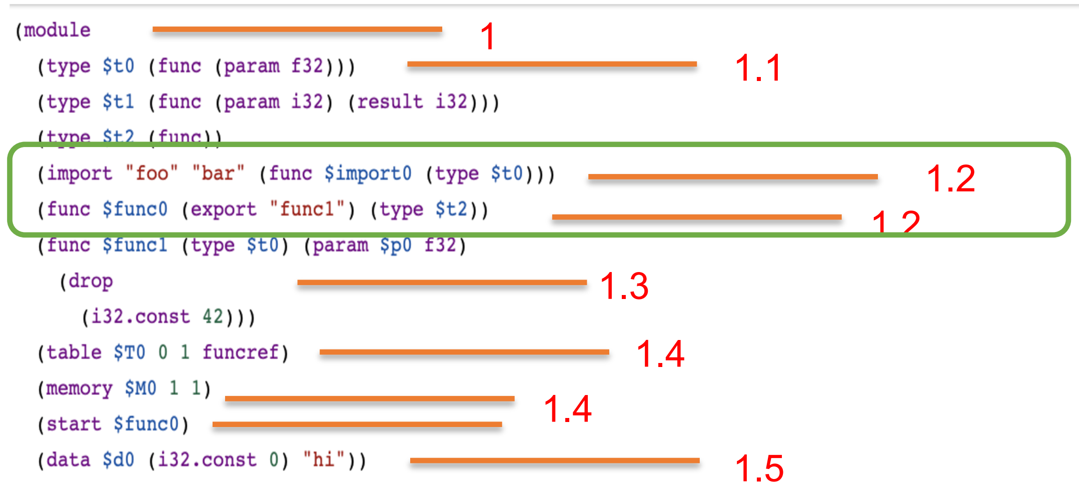
```
(module
  (type $t0 (func (param f32)))
  (type $t1 (func (param i32) (result i32)))
  (type $t2 (func))
  (import "foo" "bar" (func $import0 (type $t0)))
  (func $func0 (export "func1") (type $t2))
  (func $func1 (type $t0) (param $p0 f32)
    (drop
      (i32.const 42)))
  (table $T0 0 1 funcref)
  (memory $M0 1 1)
  (start $func0)
  (data $d0 (i32.const 0) "hi"))
```

1.2. PRE-REQUISITES

(A.K.A. LIBRARIES)

- What is the Wasm equivalent of # include <stdio.h>?
- What do we do when we **NEED** to export libraries?

Deconstructing a Wasm program



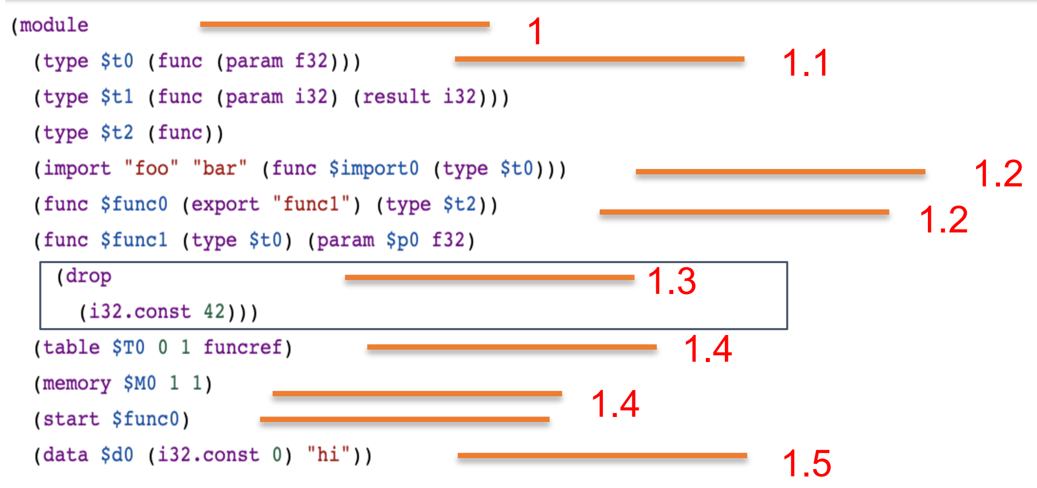
```
(module
  (type $t0 (func (param f32)))
  (type $t1 (func (param i32) (result i32)))
  (type $t2 (func))
  (import "foo" "bar" (func $import0 (type $t0)))
  (func $func0 (export "func1") (type $t2))
  (func $func1 (type $t0) (param $p0 f32)
    (drop
      (i32.const 42)))
  (table $T0 0 1 funcref)
  (memory $M0 1 1)
  (start $func0)
  (data $d0 (i32.const 0) "hi"))
```

1.3. INSTRUCTIONS

(A.K.A WHAT SHOULD THE MACHINE DO WITH YOUR PROGRAM)

- Types of instructions
    - Numeric
    - Vector
    - Reference
    - Parametric
    - Variable
    - Table
    - Memory
    - Control

Deconstructing a Wasm program



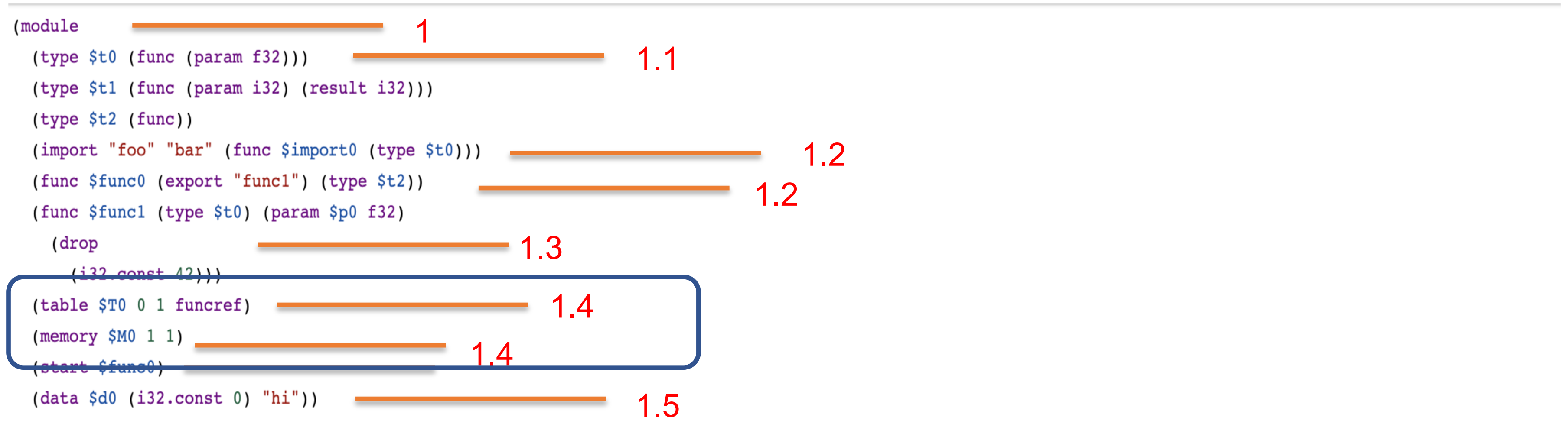
```
(module
  (type $t0 (func (param f32)))
  (type $t1 (func (param i32) (result i32)))
  (type $t2 (func))
  (import "foo" "bar" (func $import0 (type $t0)))
  (func $func0 (export "func1") (type $t2))
  (func $func1 (type $t0) (param $p0 f32)
    (drop
      (i32.const 42)))
  (table $T0 0 1 funcref)
  (memory $M0 1 1)
  (start $func0)
  (data $d0 (i32.const 0) "hi"))
```

1.5. MEMORY

(STORE?)

- Different kinds of memory?
- How do we amend the memory being allocated?

Deconstructing a Wasm program

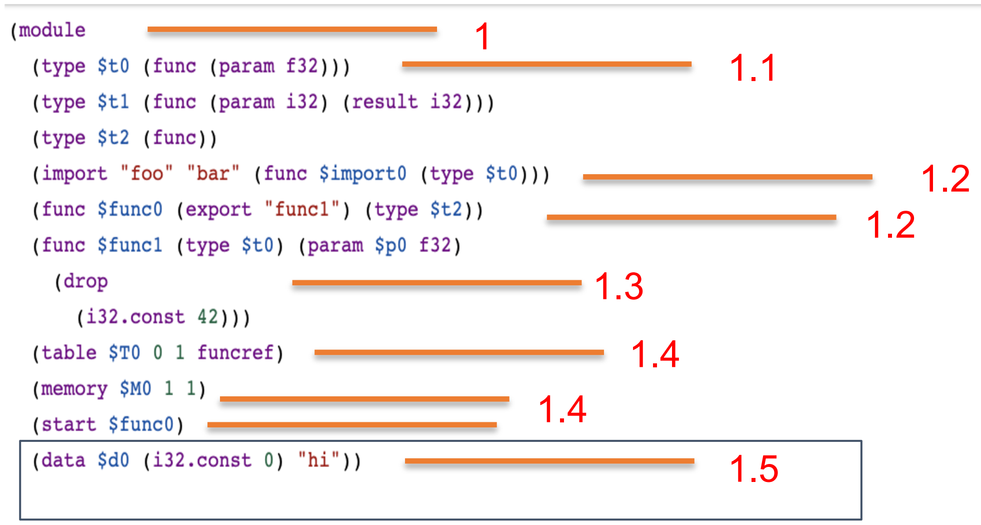


```
(module
  (type $t0 (func (param f32)))
  (type $t1 (func (param i32) (result i32)))
  (type $t2 (func))
  (import "foo" "bar" (func $import0 (type $t0)))
  (func $func0 (export "func1") (type $t2))
  (func $func1 (type $t0) (param $p0 f32)
    (drop
      (i32.const 42)))
  (table $T0 0 1 funcref)
  (memory $M0 1 1)
  (start $func0)
  (data $d0 (i32.const 0) "hi"))
```

1.6. DATA

(FOR INITIALIZATION OF MEMORY INDEX)

Deconstructing a Wasm program



```
(module
  (type $t0 (func (param f32)))
  (type $t1 (func (param i32) (result i32)))
  (type $t2 (func))
  (import "foo" "bar" (func $import0 (type $t0)))
  (func $func0 (export "func1") (type $t2))
  (func $func1 (type $t0) (param $p0 f32)
    (drop
      (i32.const 42)))
  (table $T0 0 1 funcref)
  (memory $M0 1 1)
  (start $func0)
  (data $d0 (i32.const 0) "hi"))
```

SUMMING IT ALL UP!

THANK YOU!