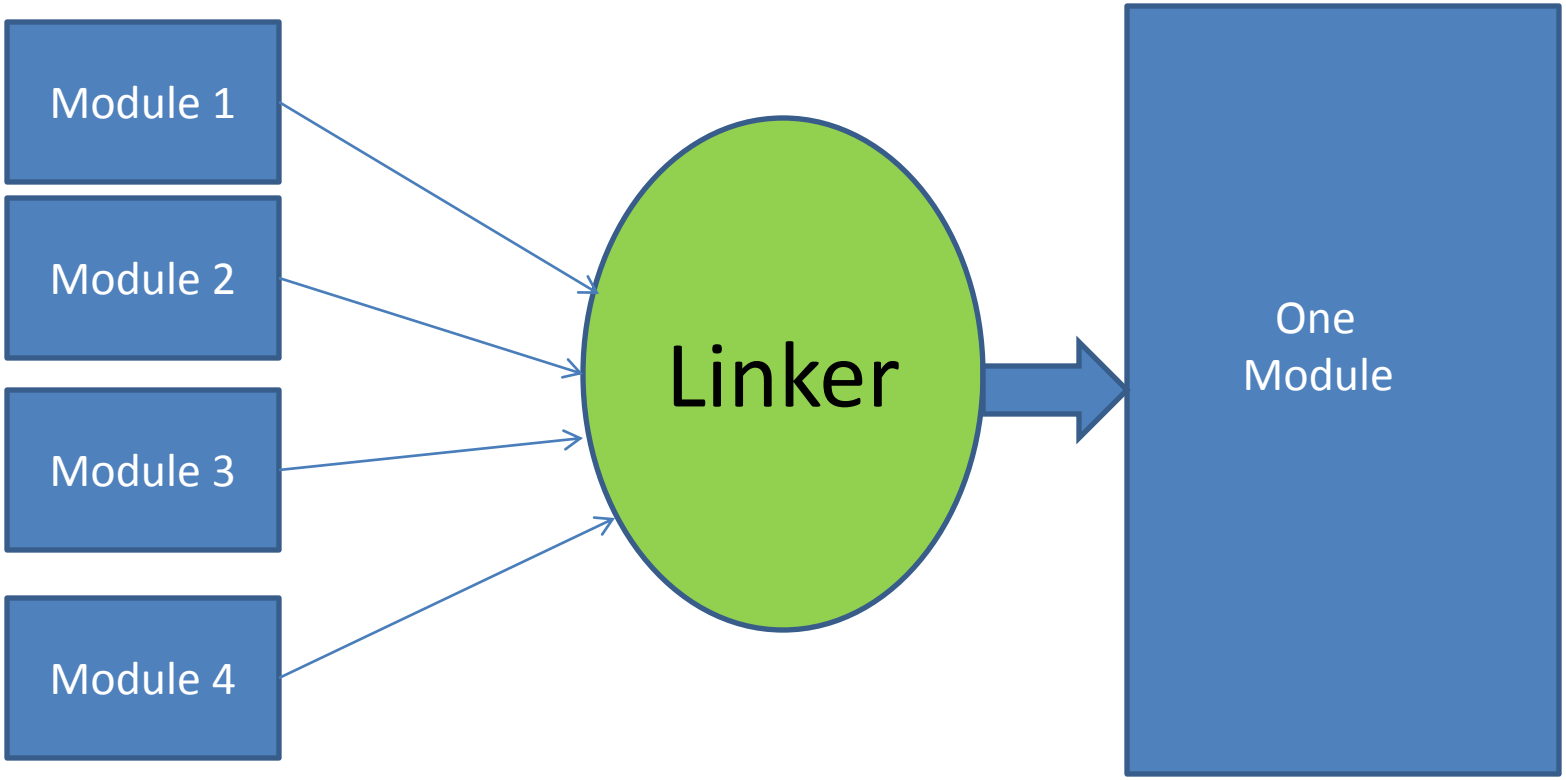
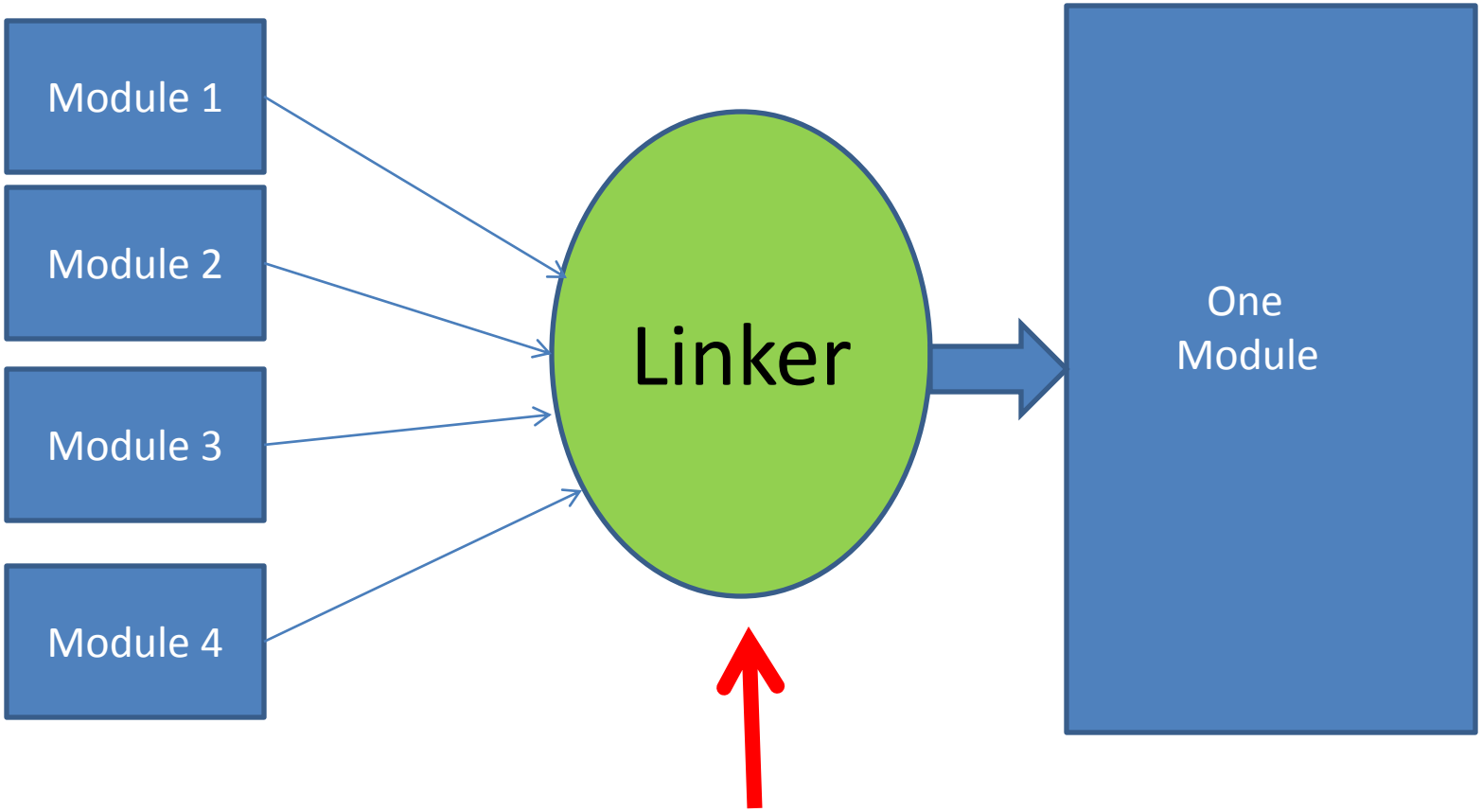
Linker

The Big Picture



The Big Picture



Two Things

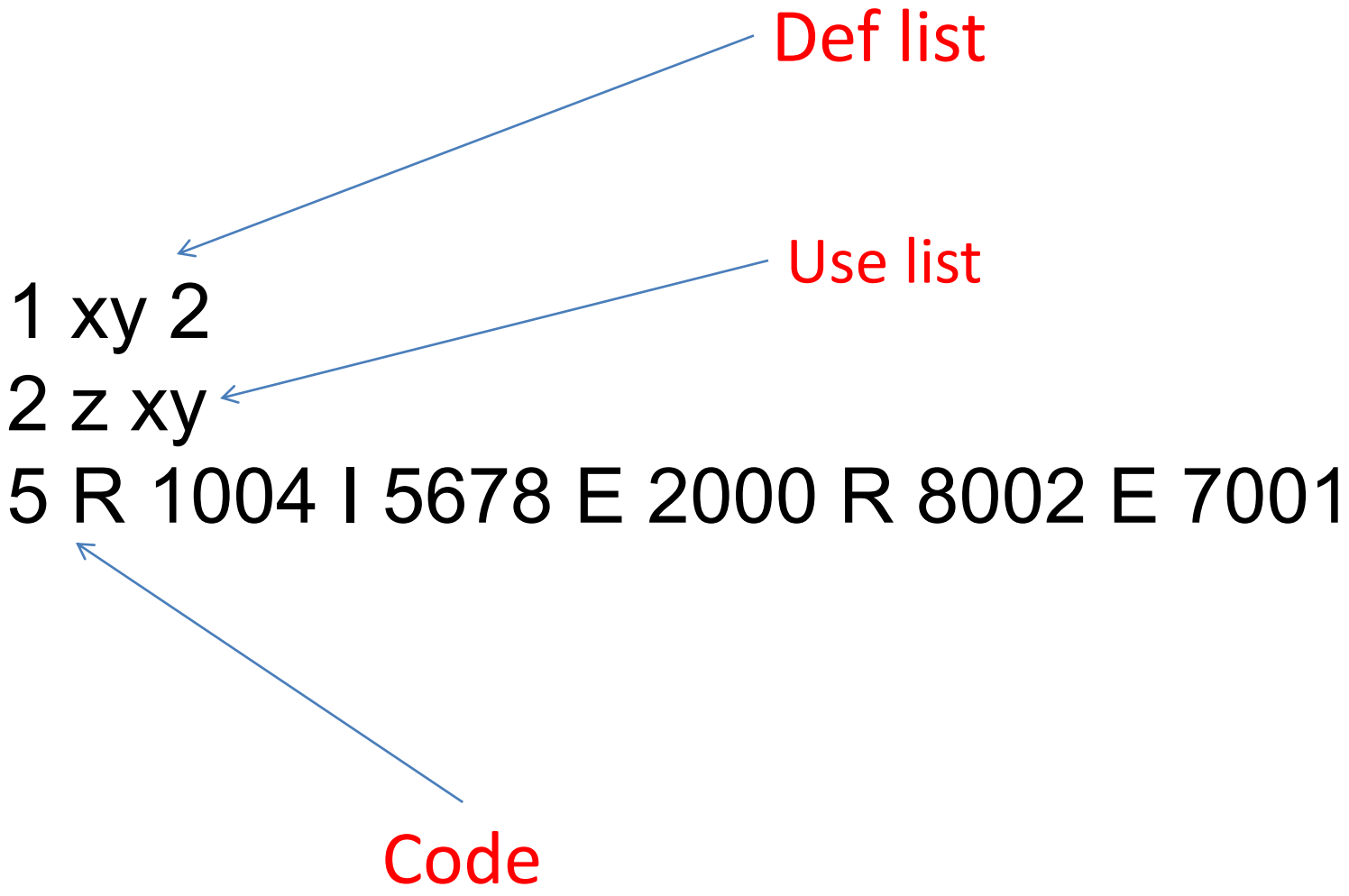
- Relocation: Translate from relative address to absolute address

- Resolve: Get the absolute address of external variables

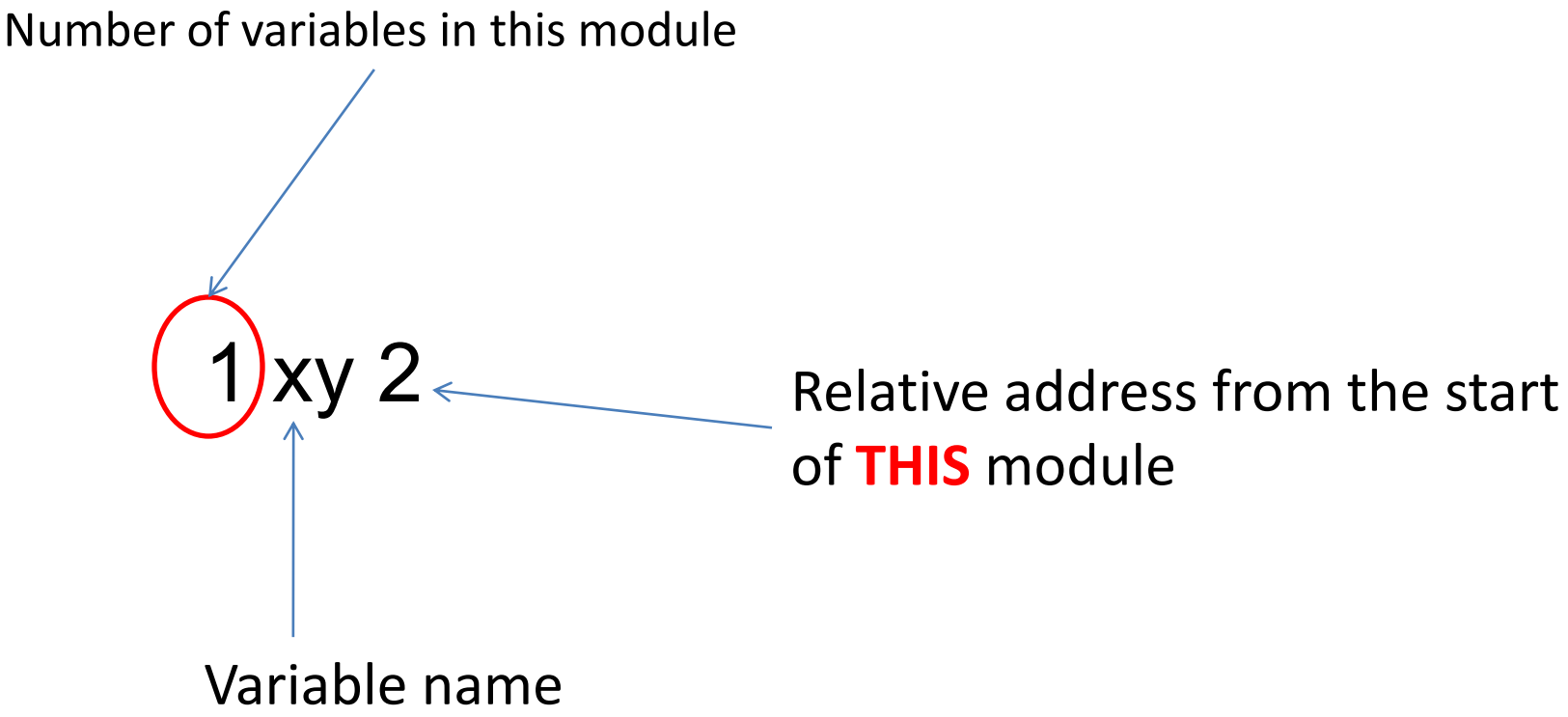
Assumptions

- A machine with memory size of 600 words
- Each word has an address (word-addressable)
- 1 word = 4 decimal digits
- Leftmost digit is opcode
- The other 3 are addresses
- Each module is divided into 3 parts
  - Definition list
  - Use list
  - Code

1 xy 2
2 z xy
5 R 1004 I 5678 E 2000 R 8002 E 7001

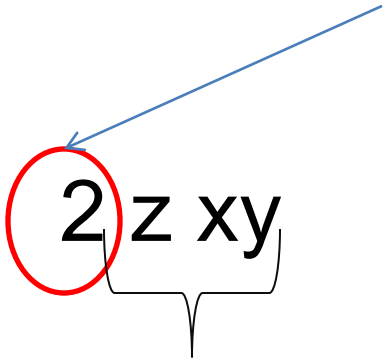

Def list
Use list
1 xy 2
2 z xy
5 R 1004 I 5678 E 2000 R 8002 E 7001
Code

Def list


Number of variables in this module
1 xy 2
Relative address from the start of THIS module
Variable name

Use list

Number of variables used in this module



List of variables (some are local and some are externals)

Code

5 R 1004 I 5678 E 2000 R 8002 E 7001

Module size

The code:

I : Immediate (unmodified by linker)

A: Absolute (unmodified by linker)

R: Relative (needs relocation)

E: External (needs to be resolved)

Code

Module size

The code:
I : Immediate (unmodified by linker)
A: Absolute (unmodified by linker)
R: Relative (needs relocation)
E: External (needs to be resolved)

The right most 3 digits designate the variable in the use list, starting from 0.
So here it means the second variable

What You Need To Do

Process all the modules twice

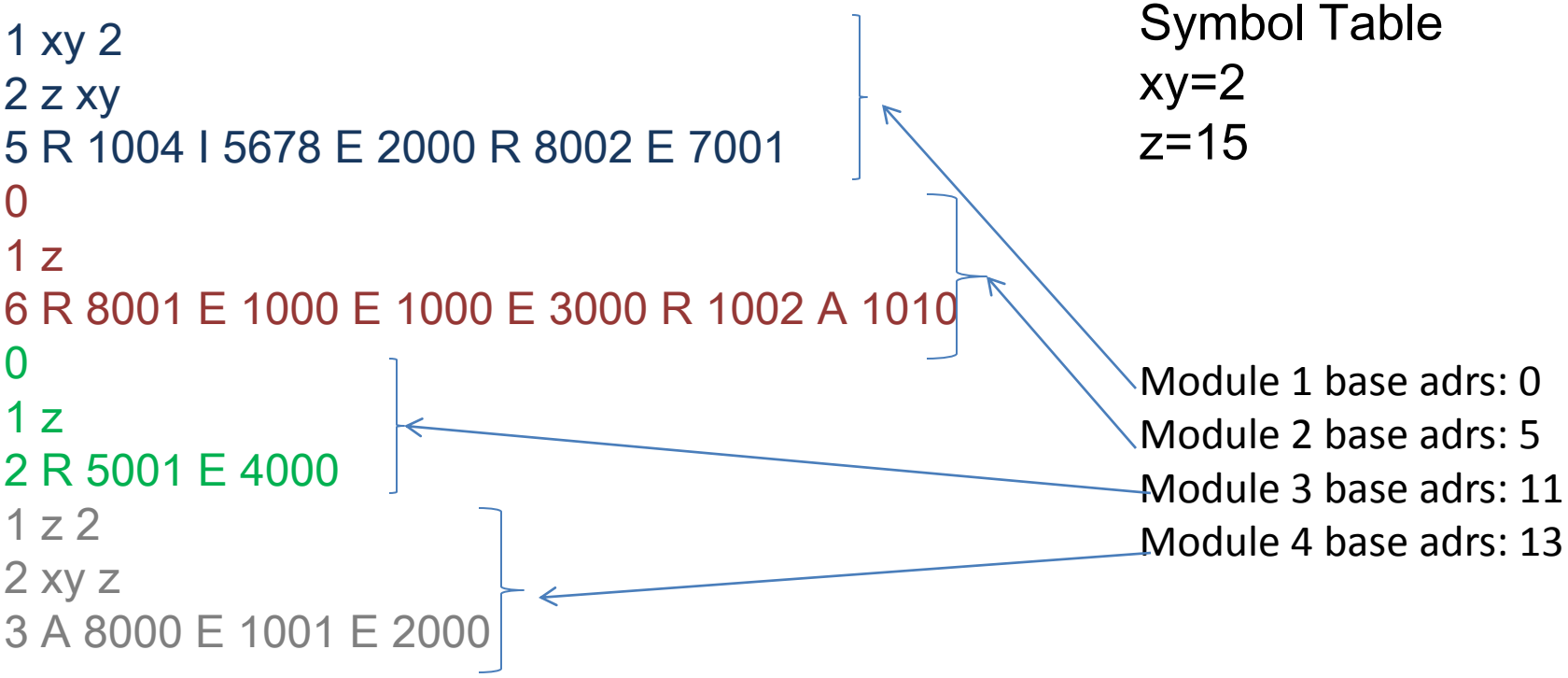
- First Pass:
  - Calculate base address for each module
  - Generate symbol table with all variables from all modules and their absolute address
- Second Pass:
  - Relocate Relative addresses (instructions with R in front of them)
  - Resolve external reference (instructions with E in front of them)

Example

1 xy 2
2 z xy
5 R 1004 I 5678 E 2000 R 8002 E 7001
0
1 z
6 R 8001 E 1000 E 1000 E 3000 R 1002 A 1010
0
1 z
2 R 5001 E 4000
1 z 2
2 xy z
3 A 8000 E 1001 E 2000

Example

After First Pass



Relocating Relative Addresses

Example

During Second Pass

```
1 xy 2
2 z xy
5 R 1004 I 5678 E 2000 R 8002 E 7001
0
1 z
6 R 8001 E 1000 E 1000 E 3000 R 1002 A 1010
0
1 z
2 R 5001 E 4000
1 z 2
2 xy z
3 A 8000 E 1001 E 2000
```

Symbol Table
xy=2
z=15

Module 1 base adrs: 0
Module 2 base adrs: 5
Module 3 base adrs: 11
Module 4 base adrs: 13

Relocating Relative Addresses

Example

During Second Pass

```
1 xy 2
2 z xy
5 R 1004 I 5678 E 2000 R 8002 E 7001
0
1 z
6 R 8006 E 1000 E 1000 E 3000 R 1007 A 1010
0
1 z
2 R 5012 E 4000
1 z 2
2 xy z
3 A 8000 E 1001 E 2000
```

Symbol Table
xy=2
z=15

Module 1 base adrs: 0
Module 2 base adrs: 5
Module 3 base adrs: 11
Module 4 base adrs: 13

Resolving Absolute Addresses

Example

During Second Pass

1 xy 2
2 z xy
5 R 1004 I 5678 E 2000 R 8002 E 7001
0
1 z
6 R 8006 E 1000 E 1000 E 3000 R 1007 A 1010
0
1 z
2 R 5012 E 4000
1 z 2
2 xy z
3 A 8000 E 1001 E 2000

Symbol Table
xy=2
z=15

Module 1 base adrs: 0
Module 2 base adrs: 5
Module 3 base adrs: 11
Module 4 base adrs: 13

Resolving Absolute Addresses

Example

During Second Pass

1 xy 2
2 z xy
5 R 1004 I 5678 E 2015 R 8002 E 7002
0
1 z
6 R 8006 E 1015 E 1015 E 3015 R 1007 A 1010
0
1 z
2 R 5012 E 4015
1 z 2
2 xy z
3 A 8000 E 1015 E 2002

Symbol Table
xy=2
z=15

Module 1 base adrs: 0
Module 2 base adrs: 5
Module 3 base adrs: 11
Module 4 base adrs: 13

Example

```
1 xy 2
2 z xy
5 R 1004 I 5678 E 2015 R 8002 E 7002
0
1 z
6 R 8006 E 1015 E 1015 E 3015 R 1007 A 1010
0
1 z
2 R 5012 E 4015
1 z 2
2 xy z
3 A 8000 E 1015 E 2002
```

Final Output

xy=2
z=15

0: 1004
1: 5678
2: 2015
3: 8002
4: 7002
5: 8006
6: 1015
7: 1015
8: 3015
9: 1007
10: 1010
11: 5012
12: 4015
13: 8000
14: 1015
15: 2002

Your Linker Must Also Catch The Following Errors

If a symbol is multiply defined, print an error message specifying the variable and exit.

If a symbol is used but not defined, print an error message specifying the value and exit.

If a symbol is defined but not used, print a warning message specifying the value and continue.

If an address appearing in a definition exceeds the size of the module, print an error message specifying the given address and the module size and exit.

If an external address is too large to reference an entry in the use list, print an error message specifying the variable and exit.

If a symbol appears in a use list but it not actually used in the module (i.e., not referred to in an E-type address), print a warning message and continue.

If an absolute address exceeds the size of the machine, print an error message specifying that address and exit.

If a relative address exceeds the size of the module, print an error specifying that address and exit.

What To Submit

- Your source code
- A readme file that specifies:
  - How to compile your code
  - How to run your code
- Put all your files in one zip file with the following name: **lastname-firstname-linker.zip**
- Email the file to the graders with subject line: linker project submission