编译运行，输入数字进入对应的部分

模式一，汇编：
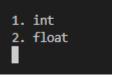

模式二，反汇编：

```
number of lines:
 >> 1
1 lines of bi code
00100010000100010000000000000100
addi $s0, $s1, 4
print enter to continue...
```

模式三，模拟运行：


```
lines of the code:
 >> 7
7 lines of bi code
00100010000100010000000000000100
00110010000100010000000000000000
00110110000100010000000000000000
00111010000100010000000000000001
00001100000000000000000000101000
00010010000100000000000001100100
00000001000000000000000000001000
```

```
                ——
          B D H
 reg 0 : 0 0 0          reg 1 : 0 0 0        reg 2 : 0 0 0        reg 3 : 0 0 0
 reg 4 : 0 0 0          reg 5 : 0 0 0        reg 6 : 0 0 0        reg 7 : 0 0 0
 reg 8 : 0 0 0          reg 9 : 0 0 0        reg 10 : 0 0 0       reg 11 : 0 0 0
 reg 12 : 0 0 0          reg 13 : 0 0 0       reg 14 : 0 0 0       reg 15 : 0 0 0
 reg 16 : 0 0 0          reg 17 : 0 0 0       reg 18 : 0 0 0       reg 19 : 0 0 0
 reg 20 : 0 0 0          reg 21 : 0 0 0       reg 22 : 0 0 0       reg 23 : 0 0 0
 reg 24 : 0 0 0          reg 25 : 0 0 0       reg 26 : 0 0 0       reg 27 : 0 0 0
 reg 28 : 0 0 0          reg 29 : 0 0 0       reg 30 : 0 0 0       reg 31 : 0 0 0
 22    11     0    4
 32    11     0    0      <<
 36    11     0    0
 3a    11     0    1
 c     0     0    28
 12    10     0    64
 2     0     0    8
 0     0     0    0
 0     0     0    0
 0     0     0    0
 0     0     0    0
 0     0     0    0
 0     0     0    0
 0     0     0    0
 0     0     0    0
 0     0     0    0
 0     0     0    0
 0     0     0    0
 0     0     0    0
 0     0     0    0
 0     0     0    0
 0     0     0    0
 0     0     0    0
 0     0     0    0

 press enter to run to next step
 []
```

按回车可以单步运行，<<处即为当前指令的位置。

模式四，int和float的转换：

```
 1. int
 2. float
 █
```

```
length :
 >> 32
fisrt :
Start your int
1. start with an int
2. start with a string
 >> 1
value :
 >> 124
second :
Start your int
1. start with an int
2. start with a string
 >> 2
string :
 >> 10101001101010
```

int支持变长，并且可以根据长度自动补全

```
first is : (123;00000000000000000000000001111011)
second is : (42149;00000000000000001010010010100101)
(123;00000000000000000000000001111011) + (42149;00000000000000001010010010100101) =
        (42272;00000000000000001010010100100000)
(123;00000000000000000000000001111011) - (42149;00000000000000001010010010100101) =
        (-42026;11111111111111110101101111010110)
(123;00000000000000000000000001111011) * (42149;00000000000000001010010010100101) =
        (5184327;00000000000000000000000000000000010011110001101101000111)
(123;00000000000000000000000001111011) / (42149;00000000000000001010010010100101) =
        (0;00000000000000000000000000000000)
press enter to continue
```

float只有32位版本并且不支持自动补全

```
first :
Start your float
1. start with an float
2. start with a string
 >> 1
value :
 >> 3.14
second :
Start your float
1. start with an float
2. start with a string
 >> 2
string :
 >> 01000001000100100110101101111011
```

```
first is : (3.14;0100000001001000111101011000010)
second is : (9.15124;01000001000100100110101101111011)
(3.14;0100000001001000111101011000010) + (9.15124;01000001000100100110101101111011) =
        (12.2912;01000001010001001010100011101011)
(3.14;0100000001001000111101011000010) - (9.15124;01000001000100100110101101111011) =
        (-6.01124;11000000110000000101110000010011)
(3.14;0100000001001000111101011000010) * (9.15124;01000001000100100110101101111011) =
        (28.7349;01000001111001011110000100010000)
(3.14;0100000001001000111101011000010) / (9.15124;01000001000100100110101101111011) =
        (0.343123;00111110101011111010110111001101)
press enter to continue
```