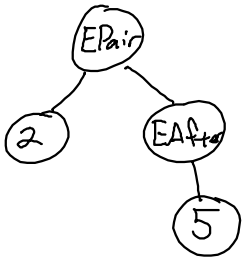
Hawk

<expr> ::= ... | (<expr>, <expr>)
| fst (<expr>)
| snd (<expr>)



(2, after(5))

⇓

(2, 6)

Tuples (pairs) won't fit in a register so use a pointer to heap. Heap stores the pair; the register value points to it.

All memory containing pairs will have addr divisible by 8. Bird pointer points to address 0x NNNN...NNN[n000] if it has value 0x NNNN...NNN[n001].

Machine values	Bird values
-2	-1
-1	true
0	0
1	pointer to 0
2	1
3	
4	2
5	
⋮	
8	4
9 = 1001	pointer to 8 = 1000

```
uint64_t bird_main(uint64_t*) asm("bird_main");
```

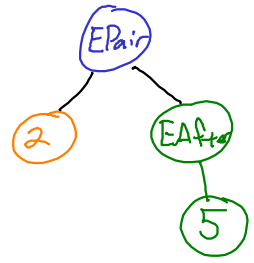
In driver

```
uint64_t* heap_ptr = malloc(....);
uint64_t result = bird_main(heap_ptr);
```

mov rdi, ...
call bird_main call 0x40750

```
.section data
  align 8
  heap_cursor:
  dq 0
.section text
  bird_main:
    push rbp
    mov rbp, rsp
    sub rsp, 16
    mov [heap_cursor], rdi
    mov rax, 4
    mov [rbp-8], rax
    mov rax, 10
    add rax, 2
    mov [rbp-16], rax
    mov rax, [heap_cursor]
    mov r10, [rbp-8]
    mov [rax], r10
    mov r10, [rbp-16]
    mov [rax+8], r10
    mov r10, [heap_cursor]
    add r10, 16
    mov [heap_cursor], r10
    add rax, 1
```

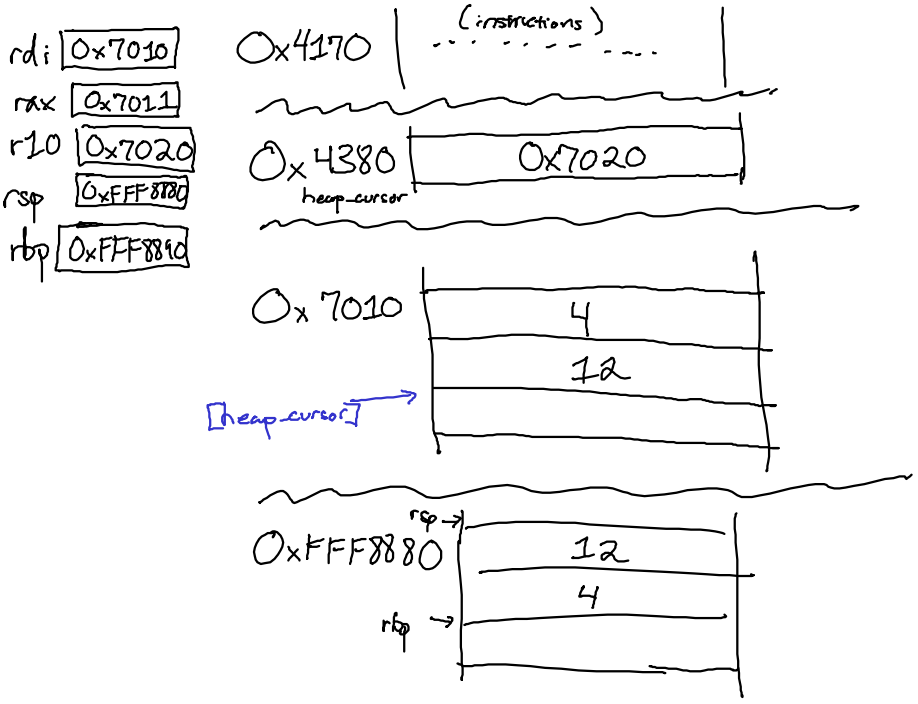
heap_cursor=0x4380



(2, after(5))

⇓

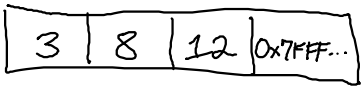
(2, 6)



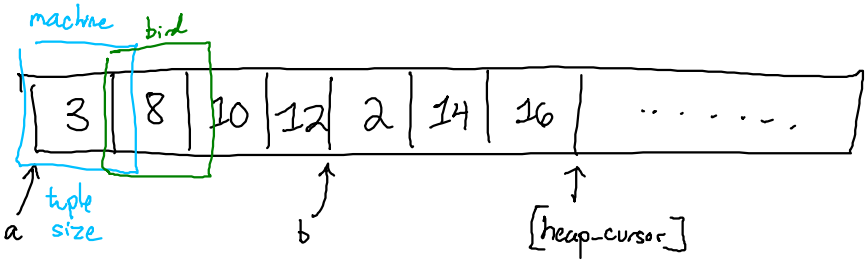
Eagle

<expr> ::= ....
| (<expr>, ..., <expr>)
| <expr> [<expr>]

(4, after(5), 6>7) ⟹ (4, 6, false)




```
let a = (4,5,6) in
let b = (7,8) in
...
```



Allocate all heap memory for a single tuple at the same time

(1, (2, 3))

incremental (1/2)