Struct

→ for pointers

- for non-pointers

if the struct variable is not a pointer (not the field)
is syntactic sugar

Suppose has field named field (\texttt{str}). field =

\texttt{Str} \rightarrow field
free DOES NOT

clear memory

```
int * iptr = (int *) malloc (30 * sizeof(int));
// iptr is used
free(iptr);
iptr[2] = 17;  // What happens?
```
// pseudo-Java

String wordList[100];

// get word from user

// insert into array

got bear cat ... 2055

to split 1

int count = 0
Better approach!

Linked List

list

cat

bear

...

zebra

golden

Call-by-value versus Call-by-reference

// in C
int swap(int a, int b) {
  int t;
  t = a;
  a = b;
  b = t;
  return t;
}
```c
int a = 6;
int b = 7;
swap(a, b);
int c = 9;
f(a, c);
```

```c
void f(int x, int y)
{
    swap(x, y);
}
```

PC = program counter

return address
instead of return address ptr

I have a call stack

activation

return addr

addr for f return

return value
\texttt{swap(a,b);}

\texttt{return a;
Call by reference

void swap(int *a, int *b)
In Java, how do you write swap?

You can't. (mostly true)
void swap(int *a, int *b) {
    int i = *a;
    *a = *b;
    *b = i;
}

main() {
    int a = 6;
    int b = 7;
    // swap(&a, &b);
    // This line is commented out.
}
int main(int argc, char **argv)

\[ a(s, t) = \begin{cases} 
31 & \text{if } a = 1.0 \text{ and } e = 1 \\
\text{other stuff} & \text{otherwise}
\end{cases} \]

\[ \text{atoi}(a) \]