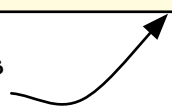


The Program's
Memory





Gets access to space
into which you can
store a value
(a record in this case)

id:	2
kind:	TXT_VAL
data.int_val:	
data.dbl_val:	Fred
data.txt_val:	

Returns the space
so that it can be allocated
to something else

Allocate Space

Free Space

Func or Proc

val: 5

Instruction: Step 2

```
Function: Read Row
-----
Returns: Row with data read from the user
Parameters:
1: next id (integer) - the id of the row to be read
Local Variables:
* line (String - 16 char max) - the text read from the user
Steps:
1: Set result's id to next id
2: Output 'Enter value: ' to the Terminal
3: Read text entered by user into line
4: If line is an integer
5:   set result's data's Int Val to the integer value in line
6:   set result's kind to INT_VAL
7: Else if line is a double
8:   set result's data's Dbl Val to the double value in line
9:   set result's kind to DBL_VAL
10: Else
11:   set result's data's Txt Val to the text in line
12:   set result's kind to TXT_VAL
13: Output "Stored in row with id ", and result's id
14: Return the result

Procedure: Main
-----
Local Variables:
* data (array containing 3 Row values)
* i (integer)
Steps:
1: for i loops over each element in data
2:   set data[i] to result of calling Read Row(i)
3: ...
```





3.1415

id:	63
kind:	DBL_VAL
data.int_val:	
data.dbl_val:	3.1415
data.txt_val:	

10.0
-5
17.21
25.1

id:	2
kind:	TXT_VAL
data.int_val:	
data.dbl_val:	
data.txt_val:	Fred

?

INT_VAL

73

Func or Proc

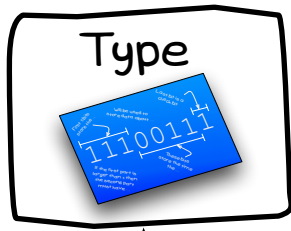
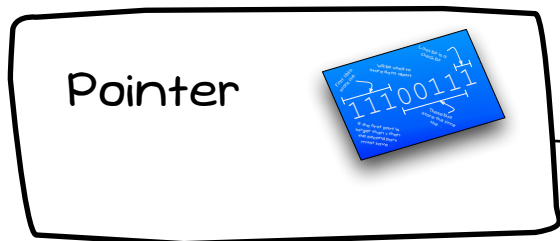
val: 5

Instruction: Step 2

```
Function: Read Row
-----
Returns: Row - a Row with data read from the user
Parameters:
1: next id (Integer) - the id of the row to be read
Local Variables:
*: line (String - 16 characters) - the text read from the user
Steps:
1: Set result's id to next id
2: Output 'Enter value: ' to the Terminal
3: Read text entered by user into line
4: If line is an integer
5:   set result's data's Int Val to the integer value in line
6:   set result's kind to INT_VAL
7: Else if line is a double
8:   set result's data's Dbl Val to the double value in line
9:   set result's kind to DBL_VAL
10: Else
11:   set result's data's Txt Val to the text in line
12:   set result's kind to TXT_VAL
13: Output "Stored in row with id ", and result's id
14: Return the result

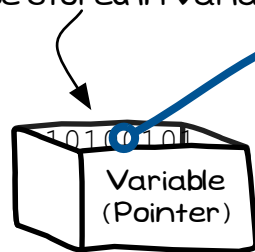
Procedure: Main
-----
Local Variables:
*: db_data (array containing 3 Row values)
*: i (Integer)
Steps:
1: for i loops over each element in db_data
2:   set db_data[i] to result of calling Read Row(i)
3: ...
```





Is a Type built into the Programming Language

Just like other types, you can have Pointer values, and these can be stored in Variables

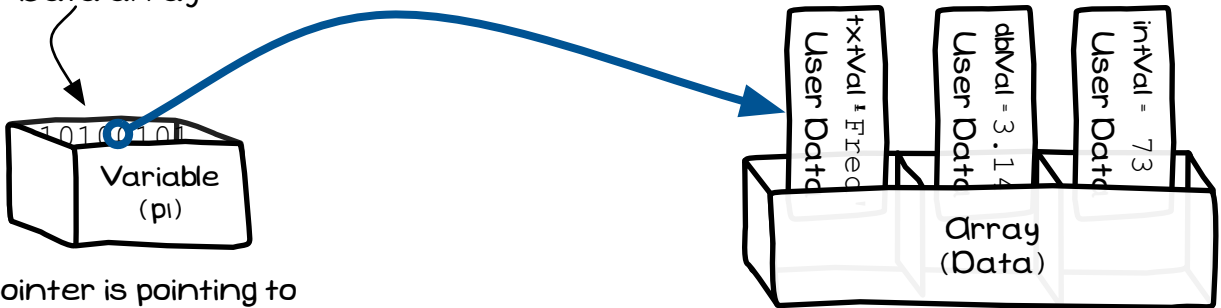


A Pointer value "points" to a value...

id:	2
kind:	TXT VAL
data.int_val:	
data.dbl_val:	Fred
data.txt_val:	

You can use the value in the pointer to access the value that it points to...

This pointer is pointing to a User Data value in the Data array



This pointer is pointing to a Point value in the Mouse variable



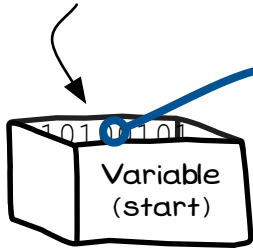
This pointer is pointing to an Integer value, the value in the x field, of the Point in the Mouse Variable.



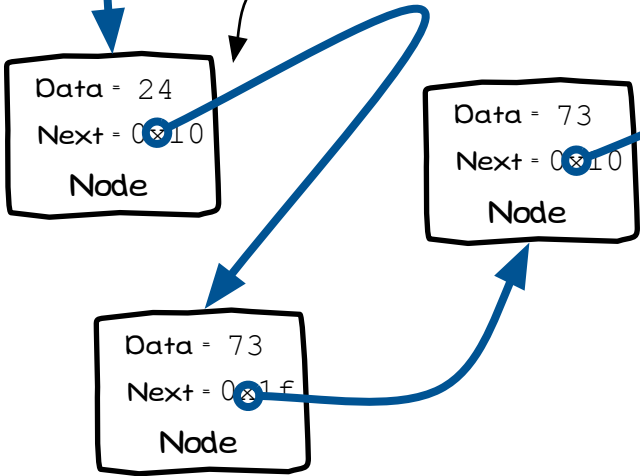
This is pointing to a Row Value record that is on the Heap



You need at least one local variable, otherwise how can you access the other values?

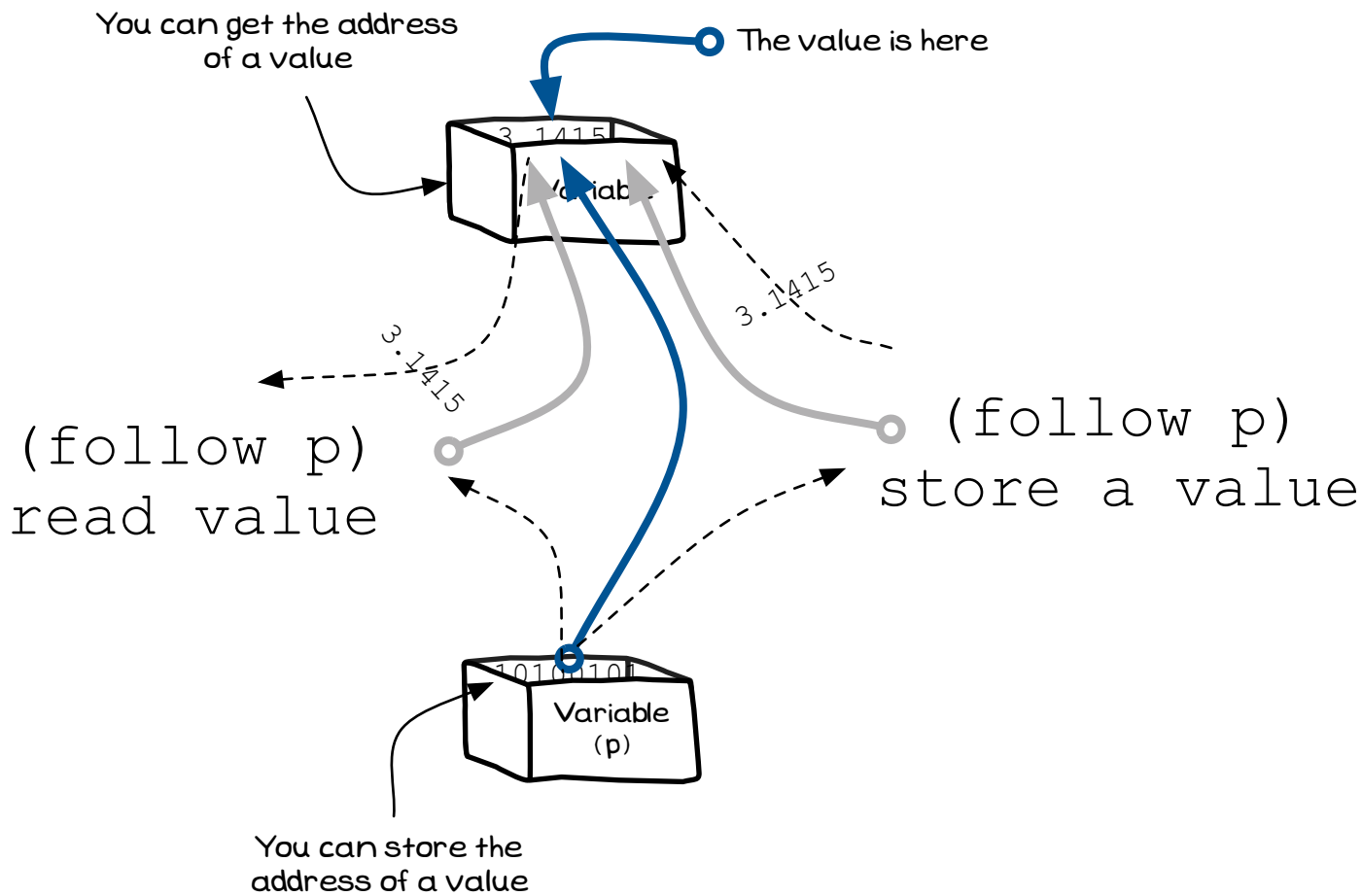


You can store pointer values on the heap, they are just like any other value...

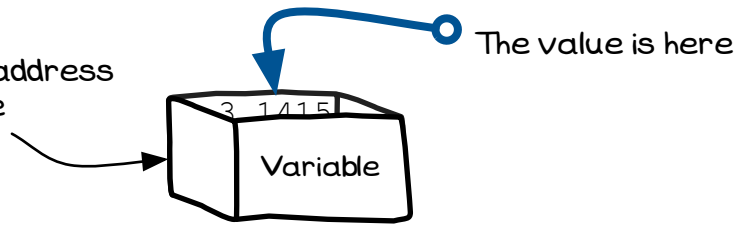


Nothing!!

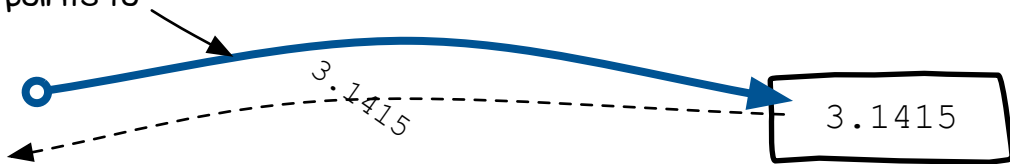
There will be a special value to represent the fact you pointer value does not point to anything...



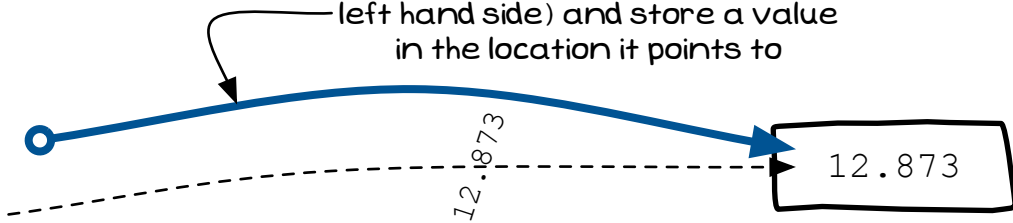
You can get the address of a value

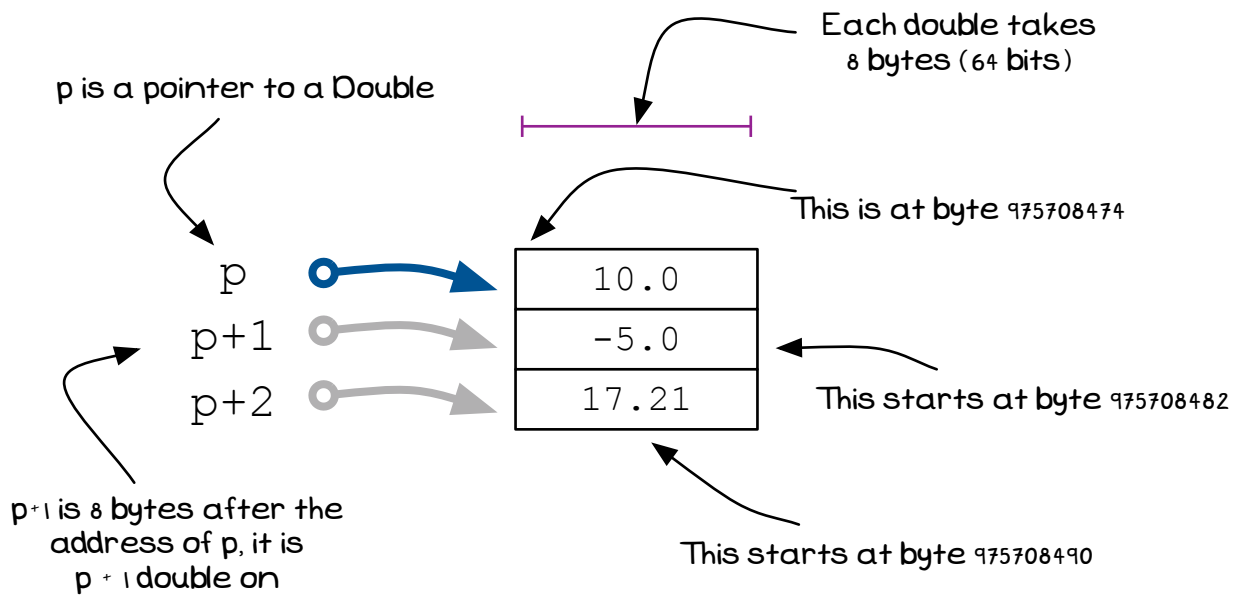


In an expression you can follow a pointer and read the value it points to



In an assignment statement you can follow the pointer (on the left hand side) and store a value in the location it points to





This value stored in p is 975708474,
the address of where the value is stored

