Data in Java:

- pointers → references, more constrained.

- Java's portability-guarantee fixes the size of all types
- No unsigned types → no conversion pitfalls
- null typically represented as "0", can't tell.
- Arrays:

C: 

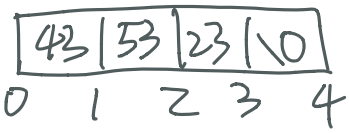
int array[4]

Java: 

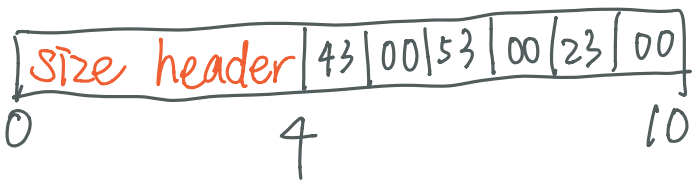
int[] arr = new int[4]

bound checking first before access

- chars & string

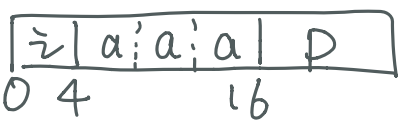
C: ASCII 

4B, 3 chars

Java: 2B Unicode 

- objects:

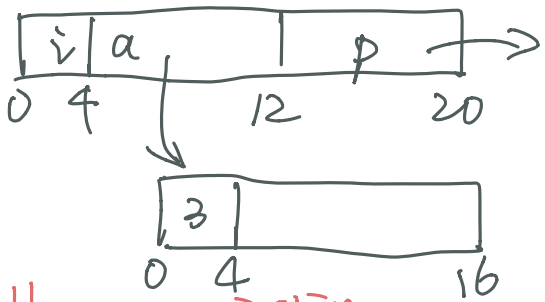
C: struct rec{};



int i int a[3] int* p

- we have "→" and "."
- pointer can point to both start and intermediate part of memory

Java: class Rec{}



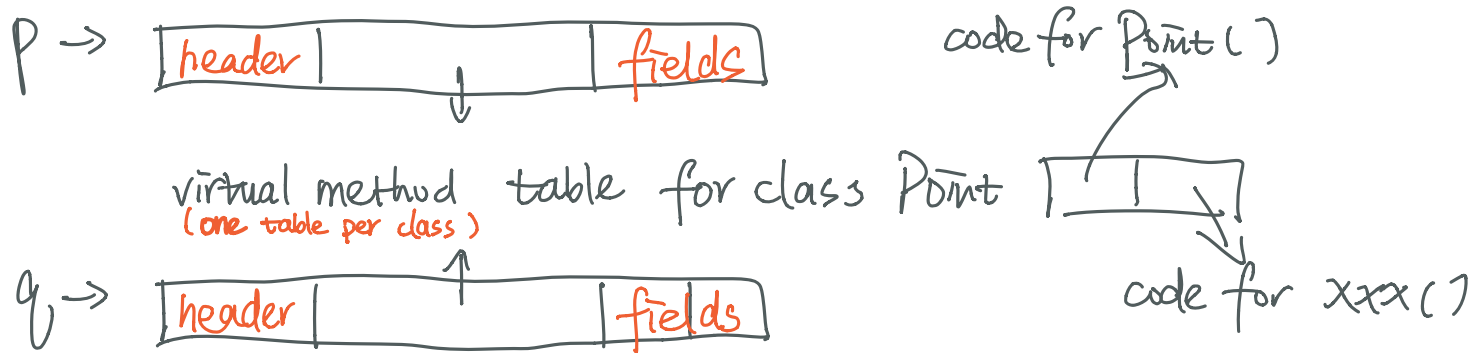
- All non-primitive vars are references to objects.

☆ No java fields need more than 8B.

- references in java only point to START of the object.

- casting in C changes dereference and arithmetic behavior
- Type-safe casting in Java can only cast compatible object references based on class hierarchy

Java Objects and Method Dispatch:



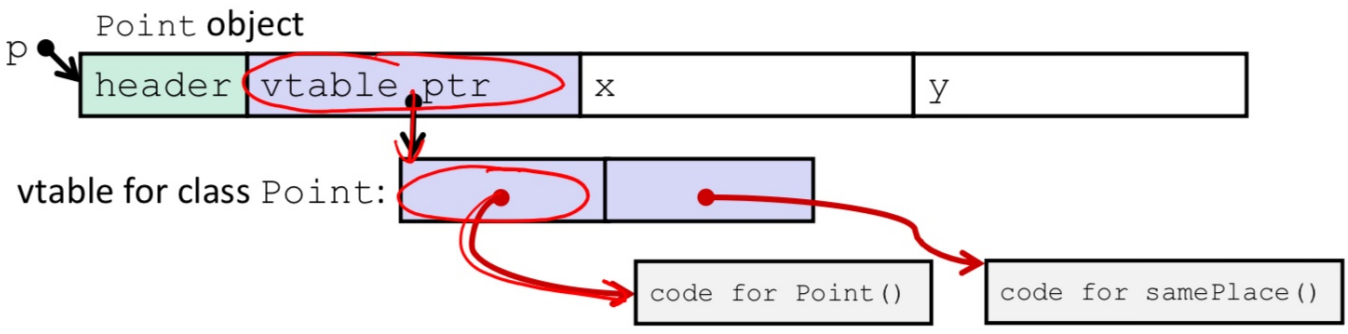
- **When we call new:** allocate space for object (data fields and references), initialize to zero/null, and run constructor method

Java:

```
Point p = new Point();
```

C pseudo-translation:

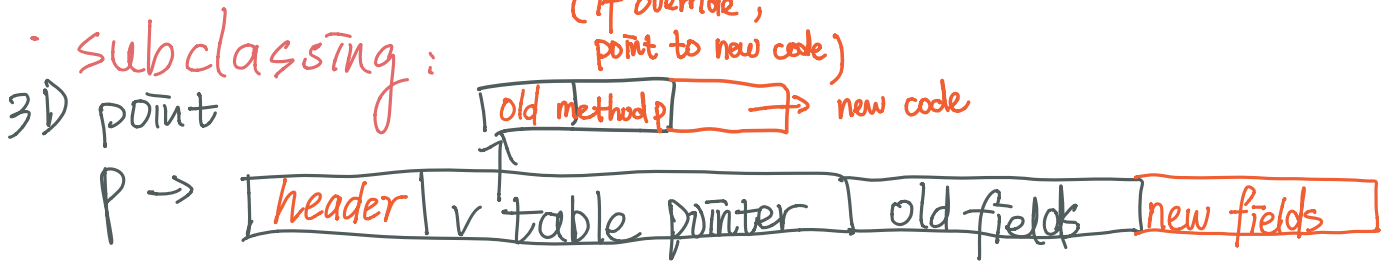
```
Point* p = calloc(1,sizeof(Point));   // zero out object data
p->header = ...;                      // set up header (somehow)
p->vtable = &Point_vtable;            // run the constructor
p->vtable[0](p);
```



- subclassing:

3D point

(if override, point to new code)



when referring to "this" → in C: p → vtable[i](p, other)
(use p → vtable)

Practice Question

- Assume: 64-bit pointers and that a Java object header is 8 B
- What are the sizes of the things being pointed at by `ptr_c` (32 B) and `ptr_j`? (44 B)

```
struct c {
  int i;
  char s[3];
  int a[3];
  struct c *p;
};
struct c* ptr_c;
```

K = 4 + 1 + 4 + 8 = 17 (internal frag)

$K_{max} = 8$ (external frag)

```
class jobj {
  int i;
  String s = "hi";
  int[] a = new int[3];
  jobj p;
}
jobj ptr_j = new jobj();
```

no explicit methods, but still inherits constructor & methods from Object class