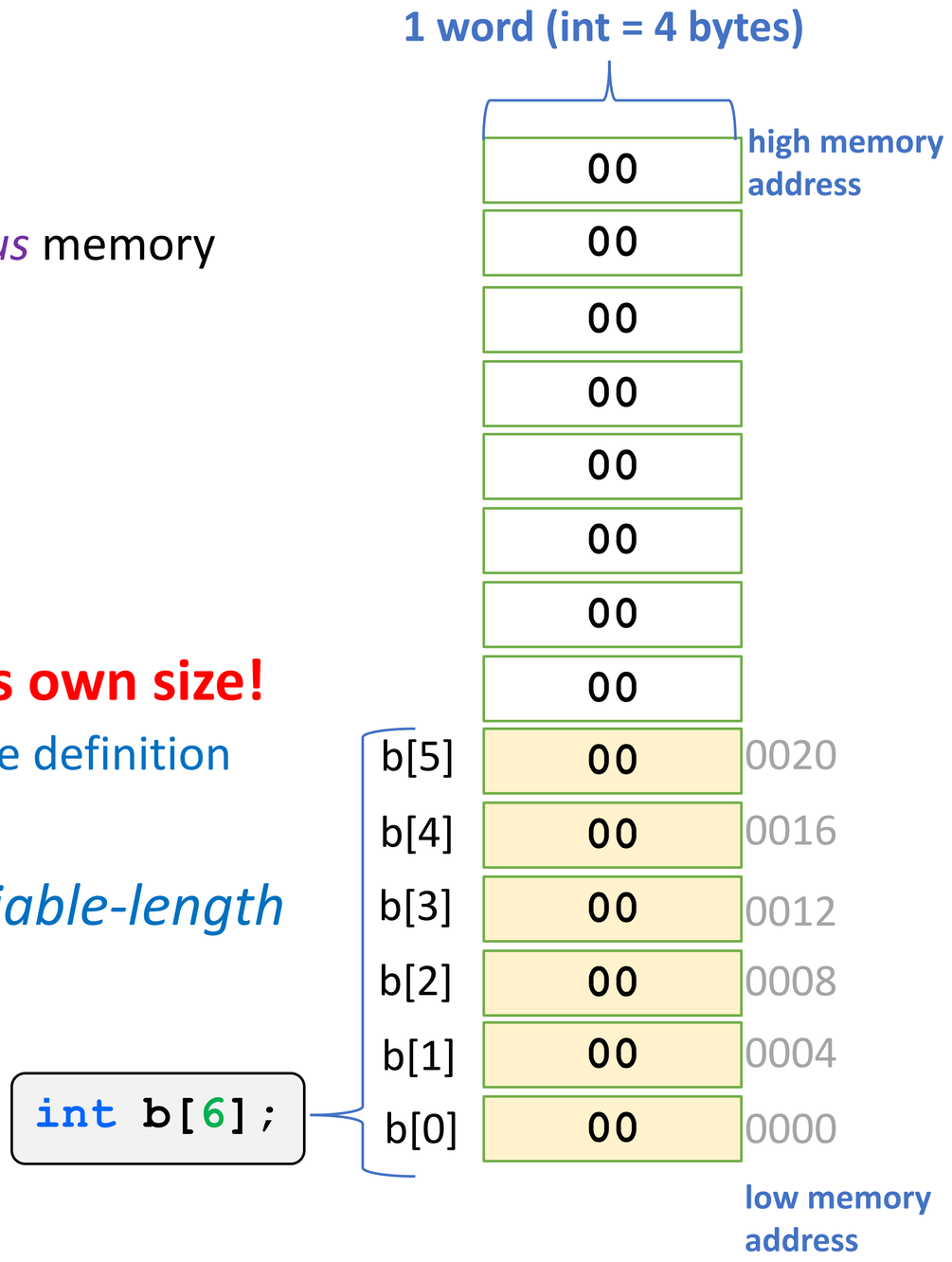
Arrays in C

- Definition: `type name[count]`
  - **Arrays are indexed starting with 0**
  - Allocates (`count * sizeof(type)`) bytes of *contiguous* memory
  - Common usage specifies compile-time constant for `count`

```
#define BSZ   6
int b[BSZ];
```

- Size of an array
  - Not stored anywhere – **an array does not know its own size!**
    - `sizeof(array)` **only works** in **scope** of array variable definition
  - Modern C versions (*not* C++) allow *automatic variable-length arrays*

```
int n = 175;
int scores[n];  // OK in C99
```



Initializing an Array in C

```
int b[5] = {2, 3, 5, 7, 11};
```

```
int b[5] = {2, 3, 5, 7, 11, 13};
```

- 13 is ignored

```
int b[] = {2, 3, 5,
           7, 11};
```

- let compiler determine the array count

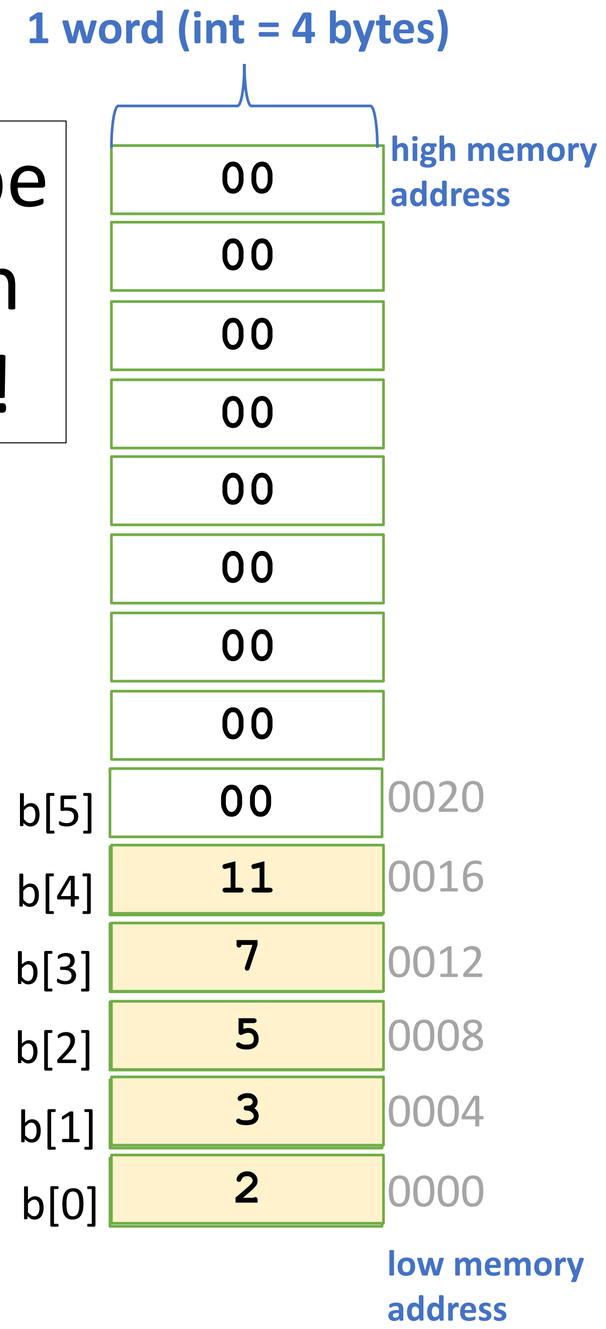
```
int arr[10] = {};
```

- fills array with 0's.

```
int arr[10];
```

- maybe initialized or not.

Arrays can be declared on the stack!!!



Working with Arrays

The size of arrays is not available readily like in Java/python. If you pass an array to a function, you also have to pass along its size.

```
int func(int [] arr, int size);
```

Arrays cannot be copied the way shown below!

```
int a[5];
int b[] = {2, 3, 5,
           7, 11};

a=b;
```