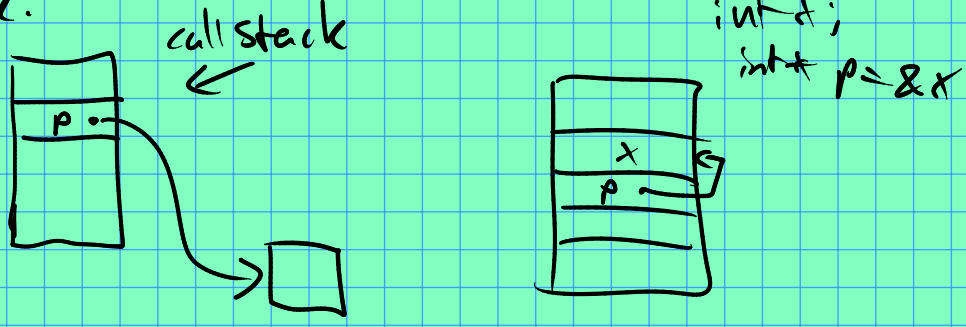
New topic: dynamic memory.

Example: dynamically allocate an integer.

```
int * p = new int.
```

New thing: "new". This is the C++ interface for dynamically allocating memory. "new" returns a pointer to the newly allocated block.



You can also allocate arrays dynamically:

```
int * A = new int[n]; // n any (positive) integer.
```

Summary of new:

1. finds contiguous block of memory of the requested size
2. gives you the address of the beginning of the block.

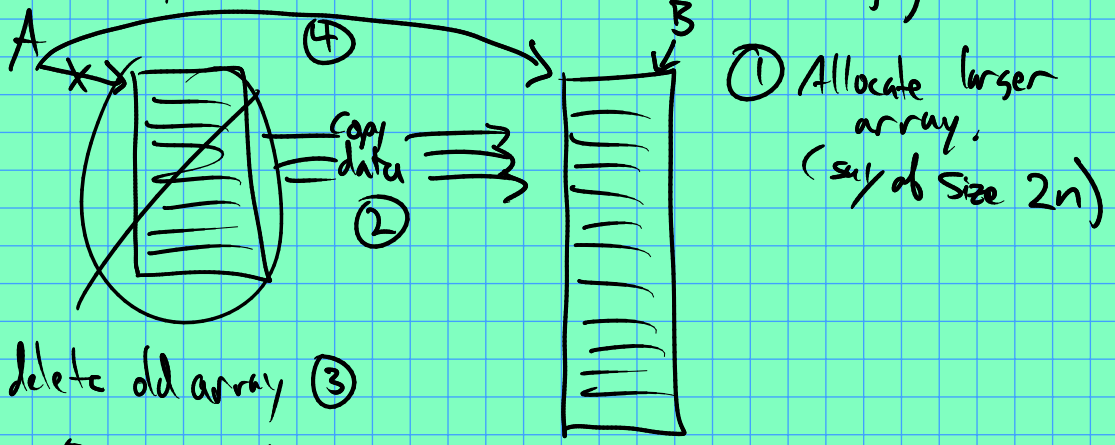
Note: you must give this memory back to the OS at some point! (when you're done with it...)

How? Use delete. E.g.,

```
delete A;
```

Example: resize a dynamically allocated array.
(might be used as a subroutine in vector when calling push_back ...)

(4) array allocated via int* A = new int[n];



①–④ in C++:

```
int* B = new int[2*n];   //① ✓
for(int i=0; i<n; i++)
    B[i] = A[i];         //② ✓
delete[] A;              //③ ✓
A = B;                   //④ ✓
```

Let's try to write our own vector.

What values would we need to keep track of?

- pointer to array of elements. (call it "data")
- size (data[0,... size-1] are valid elements of the vector.)
- capacity (≥ size) # elements in total for the data array.

keyword ↓ ↙ name of new type.

```
class vector {
    int* data;
    size_t size;
    size_t capacity;
```

```
    // also add functions...
    void push_back(int x);
    void pop_back();
};
```