Ryan Hennessey

EECS 280 Google Drive Repository
eecs280.org

EECS 280 Lab 02 Worksheet: Pointers and Arrays

Due Sunday, 28 January 2018, 8pm

1. **Fill in the final values of x, y, and z just before main returns.**
   Hint: *Draw a memory diagram in the space provided.*

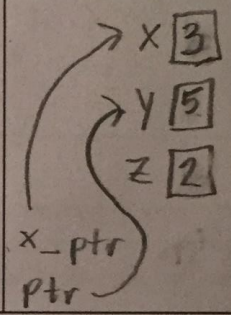
```
int main() {
  int x = 1;
  int y = 2;
  int z = 3;

  int *x_ptr = &x;
  ++(*x_ptr);

  z = *x_ptr;
  ++(*x_ptr);

  int *ptr = &y;
  *ptr = *x_ptr + z;
}
```

x	3
y	5
z	2



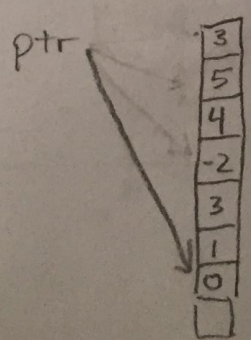
2. **What's the output of the following program? (This one is tricky!)**
   Note: *arr[] starts at address 1000*

```
int main() {
  int arr[7] = {3, 5, 4, -2, 3, 1, 0};

  int *ptr = arr;
  while (ptr < arr + 7) {
    cout << ptr << ": " << *ptr << endl;
    ptr = ptr + *ptr;
  }
}
```

Output

```
1000: 3
1003: -2
1001: 5
1006: 0
1006: 0
1006: 0
  •
  •
  •
```



1000 = arr
1001
.
.
.

1007 = arr + 7

3. **Write a function to square all the elements in the array, using traversal by index.**

```
void squareArray(int arr[], int size) {
    for(int i=0; i<size; ++i) {
        a[i] *= a[i];
    }
}
```

4. **Redo the function, using traversal by pointer instead.**

```
void squareArray(int arr[], int size){
    for(int *ptr = arr; ptr < arr+size; ++ptr) {
        *ptr *= *ptr;
    }
}
```

5. **We saw in lecture that array parameters are actually pointer parameters, which are passed by value and thus copied. However, we get behavior that makes it seem like the "whole array" was passed by reference and not copied. Briefly explain this:**

The pointer tells the program where to find the first element of the array, and since arrays are contiguous in memory, we can easily find the others as well.

6. **Draw the the state of memory as stack frames at the breakpoint labelled with a comment.**

```
int main(){
  int num1 = 42;
  int* num_ptr1 = &num1;
  int** ptr_ptr = &num_ptr1;
  int* num_ptr2 = num_ptr1;
  int num2 = 17;
  *ptr_ptr = &num2;
  ptr_ptr = &num_ptr2;
  // Breakpoint
}
```

Diagram