Lab 06: Pointers

int x=7, arr[5]={0, 10, 20, 30, 40};
For questions 1-14, the answers are each exactly **ONE** line of C++ code. Also, you may assume an previous line of code you wrote are active.

1. Declare a pointer p1 that is initially pointing at x.

   int *p1, x;

   

2. Change the value of x to 33 without using x.

   *p1 = 33;

3. Declare a pointer named p2 pointing at the first item in arr.

   int *p2 = arr;

4. Change the first item in arr to 5 without using arr.

   p2 = 5;

   

5. Change the 3rd value (the 20) in arr to 15 using [ ] and without using arr.

   p2[2] = 15;

6. Change the 4th value (the 30) in arr to 35 without using [ ] and without using arr.

   

7. Change p2 to point to the third item in arr without using arr.

   p++;

   

8. Change the item pointed at by p2 to 77 without using arr.

   

9. Write a Boolean expression that returns true iff p1 and p2 are pointing to an equivalent value.

   *p1 == *p2;

10. Write a Boolean expression that returns true iff p1 and p2 are pointing to the exact same location.

```
p1 == p2;
```

11. Make p1 point to a dynamically allocated integer (make sure to create that integer!)

```
p1 =  new int[];
```

12. Free the memory at p1.

```
delete p1;
```

13. Make p1 point to a dynamically allocated array of x integers (again, create that array!)

```
p1 =  new x[];
```

14. Free the memory at p1.

```
delete p1;
```

15) (4 points) Describe what is wrong with the following code segment, assuming it is all part of main(). Indicate if something is a compile time or runtime error or just bad programming. Also indicate how you would fix it. Line numbers have been added in ( ) for ease of reference. There are several things with the code and for full credit you must find all of them.

(1) int *p;
(2) *p=5;
(3) p=new int();
(4) *p=7;
(5) p=new int();
(6) *p=9;
(7) delete p;
(8) *p=15;

Problems:

*p in line 8 can not be used. ‘p’ was deleted in line 7.

16) (2 points) Describe the difference between statically (stack) and dynamically(heap) allocated memory. Why does C++ support **both** types?

Heap objects are allocated/deallocated dynamically as the program runs. They are prone to Memory leaks.

Consists of anything that can be completely determined at compile time. It is resizable.