Adam Stammer
Chapter 9 Book Work
CS 250

Pg. 550-553 / 2-10 even, 11, 24, 26, 28, 50, 52, 56

2.) *iptr would print out the address off x because of the & used when defining the value of iptr. ptr would not compile, because ptr was never declared. iptr would print out the address used to store the address of x.
4.) Addition and Subtraction are allowed. Division and multiplication are not.
6.) 8
8.) The program will compile if the size is not declared, or is not obviously too much. The program will crash with an out of memory error, often unseen by a user. If the size is obvious too much, newer compilers will catch it and complain with a not enough memory error.
10.) Only when the pointer is defined dynamically or outside of the function. The pointer must remain in the scope or it will be removed from memory after the function exits.

11.) A constant pointer is a pointer that can only point to its original value (storing its original address). A pointer to a constant is just a pointer that points to a value that is constant. The pointer may change, but the value may not.

24.) cout << *ptr << endl;

26.)
```
int *arr = new int[20];
for(int a = 0; a < 20; a++) {
        cout << “Enter a number for element “ << a+1 << “: “;
        cin >> arr[a];
        cout << endl;
}
//delete arr;
```

28.)
```
void getNumber(int *n) {
        cout << “Enter a number: “;
        cin >> *n;
}
```

50.) x has not been initialized and thus has no address to store in ptr.

52.) numbers + 3 must be in parenthesis to do pointer arithmetic. This will print out the third element plus three, not the third element.

56.) Val is not a pointer. *val does not dereference val, since it is not currently referenced. Removing the asterisk would multiply the value of val by two. It would be useless outside of the scope of this function, though.