//Submitted by Yunhe Liu
//CS368 HW1

1. for loop will loop through all the elements
in the array
*(array + i) refers to all the elements in the
array one by one.
*arr is always 6.
i is 0, 1, 2, 3 ... 7 in each loop
So the array will be
[6, 7, 8, 9, 10, 11, 12, 13]
after the code.

2. p = a makes p points to the first element of
the array
p++ makes p points to the second element of the array
p[4] will then give you the element that is indexed
as the 5th element (0-based indexing)
So the array will be
[1, 2, 3, 4, 5, 8, 6, 7]

3. //following is the function
//taking two InventoryItem i1 and i2
```
void stockSwap(InventoryItem i1, InventoryItem i2)
{
        //declare a temp to hold i1.widgetPtr
        Widget* temp = i1.widgetPtr;

        //assign i1.widgetPtr to i2.widgetPtr
        i1.widgetPtr = i2.widgetPtr;

        //assign i2.widgetPtr to the value stored in temp
        i2.widgetPt = temp;
}
```

4. //following is the required function
```
void setColor(Color **arr, int index, int red, int blue, int green) {
        //Color **arr is an array of pointers
        //which is color* *arr, an array of color*
        //so array[index] is a pointer pointing to
        //the struct color at index [index]

        //assign the new red value, use -> cause
        //arr[index] is a pointer but not the
```

```
        //object itself
        arr[index]->red = red;

        //assign the new blue value
        arr[index]->blue = blue;

        //assign the new green value
        arr[index]->green = green;
}
```

5. //following is the required function

```
//find node d
void moveToHead(Node head)
{
        //create two new node pointers that points to head
        Node* curr = &head;
        Node* prev = &head;

        //find d
        while(curr->value != d)
        {
                curr = curr->next
        }

        //put prev the the previous node of d
        while(prev->next->value != d)
        {
                prev = prev->next;
        }

        //relink
        //link previous of d to the next node of d
        prev->next = prev->next->next;
        //link node contains d's next to the node next to head
        curr->next = head->next;
        //link head to the node contains d
        head->next = curr;
}
```

6. Solution: 6 3 5 3

Explain:

x = 2,

y points to 4,

z points to 6;
*w is the pointer to the address of y

&x is the address of 2
*z = 6
*w = &y (the address of pointer y)