

```
:::::::::::: ptr1.cpp ::::::::::::::
#include <iostream>
using namespace std;

// ptr1.cpp -- practice using pointers
// predict output AND draw memory diagram

int f(int*, int);

int main()
{
    int    x,y;
    int    *p, *q;

    y = 2;
    x = 3;
    p = &x;
    q = p;
    *q = *p - 2;
    *p = y;
    x = f(&y, *p);
    cout << x << " " << y << " " << *p << " " << *q << endl;

    return 0;
}
int f(int* t, int u)
{
    *t = 2 * u;
    return 1 + *t;
}
:::::::::::: ptr2.cpp ::::::::::::::
#include <iostream>

using namespace std;

int main()
{
    int    x,y;
    int    *p, *q;
    int    **z;

    y = 2;
    x = 3;
    p = &x;

    q = &y;

    z = &p;

    **z = 12;
    z = &q;
    cout << **z << endl;
}
```

```
:::::::::::: ptr3.cpp ::::::::::::::
#include <iostream>
using namespace std;

// ptr3.cpp -- more practice using pointers
// predict output AND draw memory diagram

void f(string *, string *);
void show(string n[]);

int main()
{
    string n[] = { "ali", "bev", "cam", "dan", "eva", "" };
    string *p, *q;

    show(n);
    p = &n[2];
    q = p + 1;
    f(p, q);
    show(n);
    f(p-1, q+1);
    show(n);
    return 0;
}
void show(string a[])
{
    int i;
    for(i=0; a[i] != ""; i++)
        cout << a[i] << " ";
    cout << endl;
}
void f(string* p1, string *p2)
{
    string temp;
    temp = *p1;
    *p1 = *p2;
    *p2 = temp;
}
```

Put Diagrams and Output Here

```
:::::::::::: ptr4.cpp ::::::::::::::
#include <iostream>
using namespace std;

// ptr2.cpp -- more practice using pointers
// predict output AND draw memory diagram

int main()
{
    int    a[5] = { 0, 7, 2, 8, 3 };
    int    *p, *q;
    int    **z;
    int    m;

    p = &a[2];
    q = p+1;
    z = &p;
    m = *p + *q;
    ++(*z);
    ++(*p);
    cout << m << " " << **z << " " << *q << endl;
    for(int i=0; i<5; i++)
        cout << a[i] << endl;

    return 0;
}

:::::::::::: ptr5.cpp ::::::::::::::
#include <iostream>

using namespace std;

int main()
{
    int    x,y;
    int    *p, *q;
    int    **z;

    y = 2;
    x = 3;
    p = &x;
    q = p;
    *q = *p - 2;
    *p = y;
    cout << x + *p + ++*q << endl ;
    cout << " p and q are " << (unsigned long) p << " and " <<
        (unsigned long) q << endl;

    cout << "about to store a long in a pointer..." << endl;

    p = (int *) 4;

    cout << " p and q are " << (unsigned long) p << " and " <<
        (unsigned long) q << endl;

    cout << "The data stored at location 4 is: " ;
    cout << *p << endl;
}
```

```
..... revlist1.cpp .....
#include <iostream>
using namespace std;

// revlist1.cpp -- reverse a list of words
//
// input: a list of strings; output: the list in reverse
// uses: an array of strings,
// TODO: stop if no more space in array
// TODO: Write functions for each operation (readin, revprint)
// TODO: reverse all the words also
//      example: this is a test -> tset a si siht
// TODO: report if list of strings is a palindrome
// TODO: allow for any number of strings
//

const int CAPACITY = 1000;           // space in the array

struct WordList {
    string words[CAPACITY];          // storage
    int    used;                     // number used
    int    capacity;                 // total space available
};

int main()
{
    WordList list;                   // create a struct
    list.capacity = CAPACITY;        // set its properties
    list.used     = 0;

    string w;

    // read in
    while( cin >> w )
    {
        list.words[list.used] = w;
        list.used++;
    }

    // print out
    for(int i = list.used-1; i>=0 ; i-- )
    {
        cout << list.words[i] << endl;
    }
}
```