The following program does not work. Run this in the debugger. Set a breakpoint at the first marked statement. Run the program one statement at a time until you figure out what's wrong. Fix the problem and set a breakpoint at the second marked statement. Again, run one statement at a time until you figure out what's wrong. Fix the problem and run the program to completion.

```
#include <iostream>
using namespace std;

int main()
{
      int i,j;
      int Adiag[25][25];

      // create diagonalization array
      for (i=0 ; i<25 ; i++)
      {
            for (j=0 ; j<25 ; i++)
            {
                  Adiag[i][j] = i+j;
            }
      }
      // print diagonalization array
      for (i=0 ; i<25 ; i++)
      {
            for (j=0 ; i<25 ; j++)
            {
                  cout<<" "<<"Adiag[i][j];
            }
            cout<<endl;
      }
      return 0;
   }
```


List the bugs you found. What did the program do BEFORE each bug was fixed?

The following program has a bug. Set breakpoints in the debugger, and find out what the bug is.

```
#include <iostream>
using namespace std;

// function prototypes
void FillArray(int * Arr);

// create an array of 100 random numbers
int main()
{
	// make the array extra long just in case
	int * RandArray = new int[200];
	FillArray(RandArray);
	int * APtr;
	// demonstrate the use of pointers in an array loop
	for (APtr = RandArray ; RandArray != 0 ; RandArray++)
	{
		cout<<" "<<*RandArray;
	}
	return 0;
}

void FillArray(int * Arr)
{
	// set up the sentinel that indicates the end of the array
	Arr[100] = 0;
	int * Ap, i;
	// we need to count the elements - the sentinel won't work here
	for (i=0,Ap=Arr ; i<100 ; i++,Arr++)
	{
		*Ap = rand();
		// check for false sentinel and destroy it (if it occurs)
		if (*Ap == 0)
		{
			*Ap = 1;
		}
	}
}
```