Returning a local variable by reference

```
#include <iostream.h>
class Thing
{ public:
  Thing(){x =0;};
  Thing(int val) {x=val;};
  Thing(const Thing & T) {x = T.x;};
  ~Thing(){x = -1;};
  friend ostream &operator<<(ostream &, const Thing &);
  private:
    int x;
};

ostream &operator<<(ostream & output, const Thing & T)
{ output << T.x;}

Thing & f(int i)
{ Thing X(i);
 cout << "X.x = " << X << endl;
 return X; // This is THE BUG
}

main()
{ Thing A;
  cout << "A.x = " << A << endl;
  A = f(1);
  cout << "A.x = " << A << endl;
}
```

```
OUTPUT
------
A.x = 0
X.x = 1
A.x = -1
```

When returning a variable *by reference* you must be sure that the variable will exist after it returns. If the variable is local to the function, the way that X is, then it is destroyed before it can be used. This can cause core dumps.