Part 1:

The program from project 1a is split into .c and .h files called quad\_roots, lin\_root, order\_nums.

So now there are 7 files involved:

prog\_1.c

order\_nums.c

order\_nums.h

lin\_root.c

lin\_root.h

order\_nums.c

order\_nums.h

which I compile using the command: “gcc –o prog\_1 prog\_1.c order\_nums.c lin\_root.c quad\_roots.c –I.”

Find the appropriate code in the appendix.

Part 2:

and

substituting we get

we multiple both sides by . Note is necessary for this to work.

we now compare this with our initial equation that is

so we now know that

this implies

and using the other two equations we get that

diving the two we can get rid of p to get an equation we can solve for .

substituting this back into the value of p we get that

obviously this method holds the restriction that . So we must make deal with the special case when as we cannot reduce our equation to the form

That is when . Substituting we get that , .

To solve this first we consider expanding with

then substituting we get

we know that for the case where our reduction does not work. Instead we can note that the order zero term vanishes so we’re left with:

using the definition of p from earlier that reduces to

So now we know is a solution to this equation and the rest can be solved by solving the quadratic that we can later derive from it. Setting we can simply solve

(

What is this? : ?

What is this? : 15:00:36

What is this? : Feb 11 2016

**#include <stdio.h>**

**int main(void)**

**{ printf("\n What is this? : \234 ");**

**printf("\n What is this? : %s ",\_\_TIME\_\_);**

**printf("\n What is this? : %s ",\_\_DATE\_\_);**

**printf("\n \n");return(0); }**