**Chapter 5**

Names: / / .

**Roots of Polynomials**

**Section I: Linear Polynomials**

Consider the polynomial equation *ax + b = 0*; symbolically solve for the value of *x*.

*x* =

Of course the value of *x* is called the root of the linear (first degree) polynomial

*ax + b*. The TI-86 has a built-in polynomial solver package accessed via **2nd POLY** on the keyboard. This package was intentionally left off the TI-83 at the request of consulting middle and high school teachers. To find the roots of a polynomial using a CAS (Maple or Wolfram-Alpha) we can use the command

**solve(*our polynomial*=0);**

Find the roots of the polynomials in each of two ways: first "by hand", then via a CAS. Record the answers in order left to right. Remember that 2 times *x* in Maple must be entered as 2\**x*. Also the number  in Maple is Pi and  in Wolfram-Alpha is pi.

|  |  |  |  |
| --- | --- | --- | --- |
|  | HAND/Calculator | CAS | Comments |
| 1.  *x*+1 |  |  |  |
| 2.  2*x*+1 |  |  |  |
| 3. |  |  |  |
| 4.  π*x*+1 |  |  |  |
| 5.  *ax+b* |  |  |  |

Try #5 using the command **solve(*a\*x + b=0, x*);** in Maple. Why do you think the information "**,*x***" must be put into the Maple command in #5 when we didn't need it in the others?

In #5 we have encountered our first instance of using the amazing feature that most distinguishes a CAS from the previous generation of mathematics software packages: it can do Symbolic Manipulation!

**Section II. Quadratic Polynomials**

Next let's solve some quadratics. Recall that the roots of polynomials of the form *p*(*x*)=*ax*2*+bx+c* are the values of *x* which make *p*(*x*)=0. (i.e. set *p*(*x*)=0 and solve for the values of *x*.) The quadratic formula tells us the two roots of *ax*2*+bx+c* are

*x* = and *x* =

As above, find the roots of the following "by hand" and with a CAS. (Can you guess which problems will require insertion of ,x into the solve command, and which will not?)

|  |  |  |  |
| --- | --- | --- | --- |
|  | HAND | CAS | Comments |
| 6.  *x*2-1 |  |  |  |
| 7.  *x*2 |  |  |  |
| 8.  *x*2+1 |  |  |  |
| 9.  *x*2-2 |  |  |  |
| 10.  *x*2+2 |  |  |  |

What's special about #8 and #10?

Remember that (*x+a*)(*x+b*)=*x*2+(*a+b*)*x+ab*. This will be helpful in #13, 14, and 15.

|  |  |  |  |
| --- | --- | --- | --- |
|  | HAND | Maple | Comments |
| 11.  2*x*2+3*x*-21 |  |  |  |
| 12.  *x*2+*x*+1 |  |  |  |
| 13.  *x*2+(-2)*x*-2 |  |  |  |
| 14.  *x*2+(1-π)*x*-π |  |  |  |
| 15.  *x*2+(*a+b*)*x*+*ab* |  |  |  |

Now see what a CAS gives when you ask it to solve for the roots of the general/ generic quadratic polynomial

16.

*ax*2 *+ bx +c*

*x* = & *x* =

This should look familiar to you, what is it?

*Exercise:* In each of the above, 11-15, try to use the **evalf** command in Maple to obtain the HAND approximate answer. You will first need to define something to be the roots, eg.

***r*:=*expression*;**

defines *r* to be whatever is the expression. So

***r*:=solve(2*x*2-3*x*+2=0,*x*);**

defines *r* to be the two roots of 2*x*^2-3*x*+2=0.

Since r represents two roots **evalf(*r*[1]);** will evaluate the floating point approximation to the first root and **evalf(*r*[2]);** will approximate the second. So try this for each of the above. See if you can guess in advance which work and which don't.

Maple ans Approximation HAND ans evalf

|  |  |  |  |
| --- | --- | --- | --- |
| 11. |  |  |  |
| 12. |  |  |  |
| 13. |  |  |  |
| 14. |  |  |  |
| 15. |  |  |  |

Next we will play with two CAS commands that further illustrate the nice features of symbolic manipulation:

**expand(*algebraic expression*);** and **factor(*algebraic expression*);**

In each of the following first do the required calculation "by hand", then use a CAS to do it. (Remember to use the \* for multiplication in Maple.)

|  |  |  |
| --- | --- | --- |
| By hand | Expression | CAS |
|  | 17.  expand((*x*+3)(*x*-4)); |  |
|  | 18.  factor(2*x*2 +3*x* -2); |  |
|  | 19.  expand((*ax*+1)\*(*bx*+1)); |  |
|  | 20.  factor(*x*2 +(*b+a*)*x +ab*); |  |
|  | 21.  expand((*x+I*)(*x-I*)); |  |

You should now be able to "build" quadratics with any roots you like using expand and factor. Make some interesting ones.

**Section III: Higher Degree Polynomials**

In each of the following find the roots with a CAS and factor the polynomial using a CAS.

|  |  |  |
| --- | --- | --- |
| Polynomial | Roots with CAS | Factored form |
| 22.  *x* 3- *x* 2-2 *x* |  |  |
| 23.  6 *x* 3+11 *x* 2+6 *x* +1 |  |  |
| 24.  *x* 3+ *x* 2+ *x* +1 |  |  |
| 25.  *x* 4-1 |  |  |
| 26.  *x* 4+1 |  |  |
| 27.  3 *x* 4+13 *x* 3+7 *x* 2-17 *x* -6 |  |  |
| 28.  *x* 5-5 *x* 4+4 *x* 3-4 *x* 2-2 *x* +1 |  |  |

*Exercise:* We saw some quadratics and degree 4 polynomials which have only complex roots. Which numbers are they?

Experiment to see if you can find a cubic (degree 3) polynomial with only complex roots.

What is the relation between factoring a polynomial and finding its roots?

Use a CAS to solve for the zeros of the general cubic *ax*3*+bx*2*+cx+d*, assuming

*a* ≠0. They are

Why do you think students are never asked to memorize the general formula for finding the roots of a cubic polynomial?

*Exercise:* Complete the following set of Maple instructions. At each step explain just what it is you told Maple, or asked Maple to do.

>*p*:=5\**x*^3-5\**x*+1;

>solve(%=0);

>*r*:=%;

Let’s see if Maple can simplify these complicated expression gotten from that awful cubic root formula we saw above.

>simplify(*r*[1]);simplify(*r*[2]);simplify(*r*[3]);

Oh well, I guess we are stuck with them. What should we get if we substitute each of these roots into the polynomial p? Try with Maple and see what you get.

>subs(*x=r*[1],*p*);subs(*x=r*[2],*p*);subs(*x=r*[3],*p*);

See if Maple can simplify each of these expressions. What do you get?

The above expressions for the roots are the exact roots coming from that mean cubic root formula. To get something that makes more sense we can use the command evalf to get decimal approximations of them.

>evalf(*r*[1]);evalf(*r*[2];evalf(*r*[3]);

We can ask Maple to find decimal approximations of the roots directly as follows:

>fsolve(*p*=0);

How do the roots gotten from **fsolve** compare with those gotten from

approximating the exact roots?

What do you think the "f" in **fsolve** refers to?

Use a CAS to find all the roots of each of the following polynomials:

|  |  |
| --- | --- |
| Polynomial | Roots |
| 29.  *x* 2+ *x*+1 |  |
| 30.  *x* 3+ *x* 2+1 |  |
| 31.  *x* 4+ *x* 3+ *x* 2+ *x*+1 |  |
| 32.  *x* 4+ *x* 3+ *x* 2+ *x*+2 |  |