**A**

**Project Report**

**On**

**"Polynomial Solver App"**

**Prepared by**

16CE068

**Under the guidance of**

Martin K. Parmar

Assistant Professor

A Report Submitted to

Charotar University of Science and Technology

for Partial Fulfillment of the Requirements for the

3rd Semester Software Group Project-I (CE244)

**Submitted at**

****

**U. & P. U. PATEL DEPARTMENT OF**

**COMPUTER ENGINEERING**

**Chandubhai S. Patel Institute of Technology**

**At: Changa, Dist: Anand – 388421**

**Nov - Dec 2017**



**CERTIFICATE**

This is to certify that the report entitled “**Polynomial Solver App**” is a bonafied work carried out by **Mr. Patel Jainil A. (16CE068)** under the guidance and supervision of **Martin K. Parmar** for the subject **Software Group Project-I** (CE244) of 3rd Semester of Bachelor of Technology in **U & P U. Patel Department of Computer Engineering** at Faculty of Technology & Engineering – CHARUSAT, Gujarat.

To the best of my knowledge and belief, this work embodies the work of candidate himself, has duly been completed, and fulfills the requirement of the ordinance relating to the B.Tech. Degree of the University and is up to the standard in respect of content, presentation and language for being referred to the examiner.

|  |  |
| --- | --- |
| Martin K. Parmar  Assistant Professor  U & P U. Patel Department of Computer Engineering,  CSPIT, Changa, Gujarat. |  |
| Dr.(Prof.) Amit Ganatra  Head,  U & P U. Patel Department of Computer Engineering, CSPIT,  Dean,  Faculty of Technology & Engineering,  CHARUSAT, Changa, Gujarat. | |

**Chandubhai S. Patel Institute of Technology**

At: Changa, Ta. Petlad, Dist. Anand, PIN: 388 421. Gujarat

**Acknowledgement**

Knowledge in itself is a continuous process. At this moment of our substantial enhancement, I rarely find words to express our gratitude towards those who were constantly involved with me.

The completion of any inter disciplinary project depends upon coordination, cooperation and combined efforts of several resources of knowledge, creativity, skill, energy and time. The work being accomplished now, I feel our most sincere urge to recall and knowledge through these lines, trying our best to give full credits wherever it deserves.

I would like to thank our project guide **Mr. Martin K. Parmar** and Dean and Principle **Dr. Amit Ganatra** who advised and gave us moral support through the duration of project. Without there constant encouragement we could not achieve what we have.

It’s our good fortune that I had support and well wishes of many. I am thankful to all and those whose names are forgotten to acknowledge here but contributions have not gone unnoticed.

With Sincere Regards

Patel Jainil A.(16CE068)

* **Abstract**

Polynomial is equation like 3x2+2x+9 having coefficients and degree. Multiplication of large polynomial is very tedious job so app would help to solve large polynomials. App is calculator that calculates the results quickly and helpful for students. This can be used in many fields of science and mathematics to develop new concepts and derive equations.

**Index**

**Title**

1. **Description…………………………………………………………7**
2. **Software and hardware requirements……………………………..9**
3. **Major functionality………………………………………………..11**
4. **Flow chart………………………………………………………....13**
5. **Screenshots……………………………………………………..…15**
6. **Limitations of project……………………………………….……19**
7. **Outcome………………………………………………….……….21**
8. **Future Enhancement………………………………….…………28**
9. **Reference…………………………………………….…………... 30**
10. **Project Applications…………………………………………........32**

* **Project Profile**

|  |  |
| --- | --- |
| **Project Title** | Polynomial Solver |
| **Type of project** | Android Application |
| **Definition** | Solve Polynomials with one variables and to multiply and divide them. |
| **Purpose and Objective** | This useful app allows us to solve polynomials of any degree and to get fast and accurate result. User can enter only coefficients and no need to enter degree of each coefficient. |
| **Team Size** | 1 |
| **Team Member** | Patel Jainil A. |
| **Front End** | Android |
| **Back End** | Netbeans java. |
| **Tools Used** | Android Studio |
| **Functionality** | 1. Add two polynomials 2. Subtract two polynomials 3. Multiply two polynomials 4. Divide two polynomials 5. Get remainder of the division |
| **Project Guide** | Martin K. Parmar |

Project

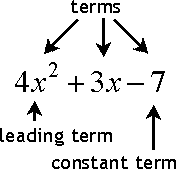
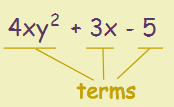
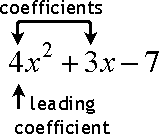
Definition

This Project is polynomial solver app, in which new algorithms are created to multiply and divide polynomials with one variable.

Algorithms are not found at internet and are unique.

We deliver services like Sudoku solver and feedback services. The project is for reducing time consumed in looping processes in mathematics.

**What is a polynomial?**

In mathematics, a **polynomial** is an expression consisting of variables (or indeterminates) and coefficients, that involves only the operations of addition, subtraction, multiplication, and non-negative integer exponents of variables. An example of a polynomial of a single indeterminate *x* is *x*2 − 4*x* + 7. An example in three variables is *x*3 + 2*xyz*2 − *yz* + 1.

Polynomials appear in a wide variety of areas of mathematics and science. For example, they are used to form polynomial equations, which encode a wide range of problems, from elementary word problems to complicated problems in the sciences; they are used to define **polynomial functions**, which appear in settings ranging from basic chemistry and physics to economics and social science; they are used in calculus and numerical analysis to approximate other functions. In advanced mathematics, polynomials are used to construct polynomial rings and algebraic varieties, central concepts in algebra and algebraic geometry.

A *polynomial function* is a function that can be defined by evaluating a polynomial. A function *f* of one argument is thus a polynomial function if it satisfies.

{\displaystyle f(x)=a\_{n}x^{n}+a\_{n-1}x^{n-1}+\cdots +a\_{2}x^{2}+a\_{1}x+a\_{0}} 

for all arguments *x*, where *n* is a non-negative integer and *a*0, *a*1, *a*2, ..., *an* are constant coefficients.

For example, the function *f*, taking real numbers to real numbers, defined by

{\displaystyle f(x)=x^{3}-x} is a polynomial function of one variable.

## **Degree**

The **degree** of a polynomial with only one variable is the **largest exponent** of that variable.

Description

* Description

Project is about taking dynamic input and manipulating polynomials efficiently. Creating loops and avoiding recursion is good for programming.

Main aim is to reduce time of users.

So only coefficients are taken with space as delimeter

App benefits

1)Dynamic

2)Enter only coefficients

3)Quick result

4)No recursion used

* I tried to make a calculator but there are many android calculators app available so decided to try Something new.
* So decision to make a calculator that takes two polynomial of degree n and degree m and multiply and divide those two polynomial was taken.
* When we get a polynomial of degree 7 and are asked to multiply with degree 6 polynomial then it is tedious job and there is no shortcut or app or directly made algorithm available. It takes much time.so a new algorithm and app can make life easier.
* App also offers a Sudoku solver for those who give feedback and ratings.

Front End

**Android Studio** is the official ntegrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems. It is a replacement for the Eclipse Android Development Tools (ADT) as primary IDE for native Android application development.

Features of android studio:

* Gradle-based build support
* Android-specific refactoring and quick fixes
* Lint tools to catch performance, usability, version compatibility and other problems
* ProGuard integration and app-signing capabilities
* Template-based wizards to create common Android designs and components
* A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations
* Support for building Anrdroid Wear apps
* Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine
* Android Virtual Device (Emulator) to run and debug apps in the Android studio.

Back end

**NetBeans** is a software development platform written in Java. The NetBeans Platform allows applications to be developed from a set of modular software components called *modules*. Applications based on the NetBeans Platform, including the NetBeans integrated development environment (IDE), can be extended by third party developers.The NetBeans IDE is primarily intended for development in Java, but also supports other languages, in particular PHP, C/C++ and HTML5.NetBeans is cross-platform and runs on Microsoft Windows, macOS, Linux, Solaris and other platforms supporting a compatible JVM.The editor supports many languages from Java, C/C++, XML and HTML, to PHP, Groovy, Javadoc, JavaScript and JSP. Because the editor is extensible, you can plug in support for many other languages.The NetBeans

Netbeans is used for functions testing at java level.

Software and hardware requirements

* Software requirement:
  + 1. Android operating system API 19 Android 4.4 (Kitkat)
    2. App:- Polynomial
* Hardware requirement:

1. Android Smart phone

Class diagram

Class Term

1. Public float coef; //coefficients
2. Public int exp; //exponents
3. Term sum(term t2)
4. Term sub(term t2)
5. Term mul(term t2)

Class Polynomial

1. Public int d;//degree of polynomial
2. Public Term t[d+1]; //terms in polynomial
3. Polynomial (int s) //constructor
4. Void getdata(int b[]) //getting data
5. String putdata() //for returning string to print
6. Polynomial sum(Polynomial p2)
7. Polynomial sub(Polynomial p2)
8. Polynomial mul(Polynomial p2)
9. Polynomial div(Polynomial p2)
10. Polynomial remainder(Polynomial p2)
11. Polynomial div1(Polynomial p2) //division if degree is 0

Major functionality

**Major Functionality**

1. **Add two polynomials**
2. **Subtract two polynomials**
3. **Multiply two polynomials**
4. **Divide two polynomials**
5. **Get remainder of the division**

**Minor Functionality**

1. **Get help of polynomials**
2. **Give feedback**
3. **Rate the app**
4. **Try new Sudoku solver.**

**Add two polynomials.**

Polynomial sum(Polynomial p2)  
{  
 **int** q;  
 **if**( **d**>p2.**d**)  
 { q= **d**; }  
 **else** { q=p2.**d**; }  
 Polynomial p3=**new** Polynomial(q);  
 **if**( **d**>p2.**d**)  
 { **for**(**int** i=q;i>=q-p2.**d**-1;i--)  
 {  
 p3.**t**[i].**coef**= **t**[i].**coef**;  
 }  
 **for**(**int** i=p2.**d**;i>=0;i--)  
 {  
 p3.**t**[i].**coef**= **t**[i].**coef**+p2.**t**[i].**coef**;  
 } }  
 **else** { **for**(**int** i=q;i>=**d**;i--)  
 {  
 p3.**t**[i].**coef**=p2.**t**[i].**coef**;  
 }  
 **for**(**int** i=**d**;i>=0;i--)  
 {  
 p3.**t**[i].**coef**= **t**[i].**coef**+p2.**t**[i].**coef**;  
 } }  
 **return** p3;  
}

**Subtract two polynomials.**

Polynomial sub(Polynomial p2)  
{  
 **int** q;  
 **if**(**d**>p2.**d**)  
 { q=**d**; }  
 **else** { q=p2.**d**; }  
 Polynomial p3=**new** Polynomial(q);  
 **if**( **d**>p2.**d**)  
 {  
 **for**(**int** i=q;i>=q-p2.**d**-1;i--)  
 {  
 p3.**t**[i].**coef**= **t**[i].**coef**;  
 }  
 **for**(**int** i=p2.**d**;i>=0;i--)  
 {  
 p3.**t**[i].**coef**= **t**[i].**coef**-p2.**t**[i].**coef**;  
 }  
 }  
 **else** {  
 **for**(**int** i=q;i>=**d**;i--)  
 {  
 p3.**t**[i].**coef**=p2.**t**[i].**coef**;  
 }  
 **for**(**int** i=**d**;i>=0;i--)  
 {  
 p3.**t**[i].**coef**= **t**[i].**coef**-p2.**t**[i].**coef**;  
 }  
 }  
 **return** p3;  
  
}

**Multiply two polynomials.**

Polynomial mul(Polynomial p2)  
{  
 **int** x;  
 x= **d**+p2.**d**;  
 Polynomial p3=**new** Polynomial(x);  
 **for**(**int** i= **d**;i>=0;i--)  
 {  
 **for**(**int** j=p2.**d**;j>=0;j--)  
 {  
 p3.**t**[i+j]=p3.**t**[i+j].sum( **t**[i].mul(p2.**t**[j]));  
 }  
  
 }  
 **return** p3;  
}

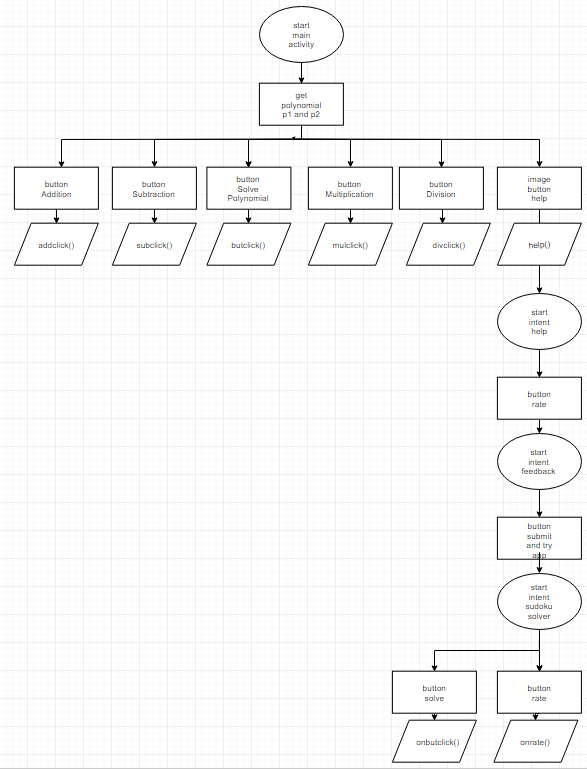
**Divide two polynomials.**

Polynomial div(Polynomial p2)  
{  
 Polynomial p3=**new** Polynomial(p2.**d**-**d**);  
 Polynomial p4=**new** Polynomial(p2.**d**-1);  
 **int** i=p2.**d**;  
 **for**(**int** j=p2.**d**-**d**+1;j>0;j--)  
 {  
 **float** f=p2.**t**[i].**coef**/**t**[**d**].**coef**;  
 p3.**t**[j-1].**coef**=f;  
 p4.**t**[j-1].**coef**=f;  
 p2=p2.sub(**this**.mul(p4));  
 p4.**t**[j-1].**coef**=0;  
 i--;  
 }  
 **return** p3;  
}

**Remainder of two polynomials**

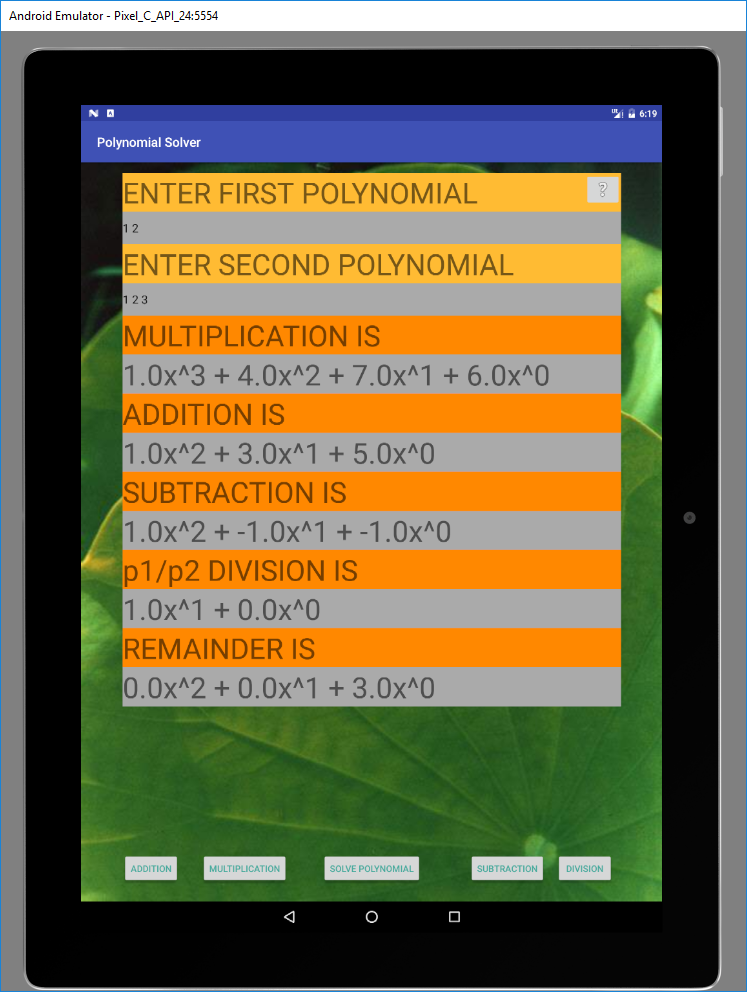
Polynomial remainder(Polynomial p2)  
{  
 Polynomial p3=**new** Polynomial(p2.**d**-**d**);  
 Polynomial p4=**new** Polynomial(p2.**d**-1);  
 **int** i=p2.**d**;  
 **for**(**int** j=p2.**d**-**d**+1;j>0;j--)  
 {  
 **float** f=p2.**t**[i].**coef**/**t**[**d**].**coef**;  
 p3.**t**[j-1].**coef**=f;  
 p4.**t**[j-1].**coef**=f;  
 p2=p2.sub(**this**.mul(p4));  
 p4.**t**[j-1].**coef**=0;  
 i--;  
 }  
 **return** p2;  
}

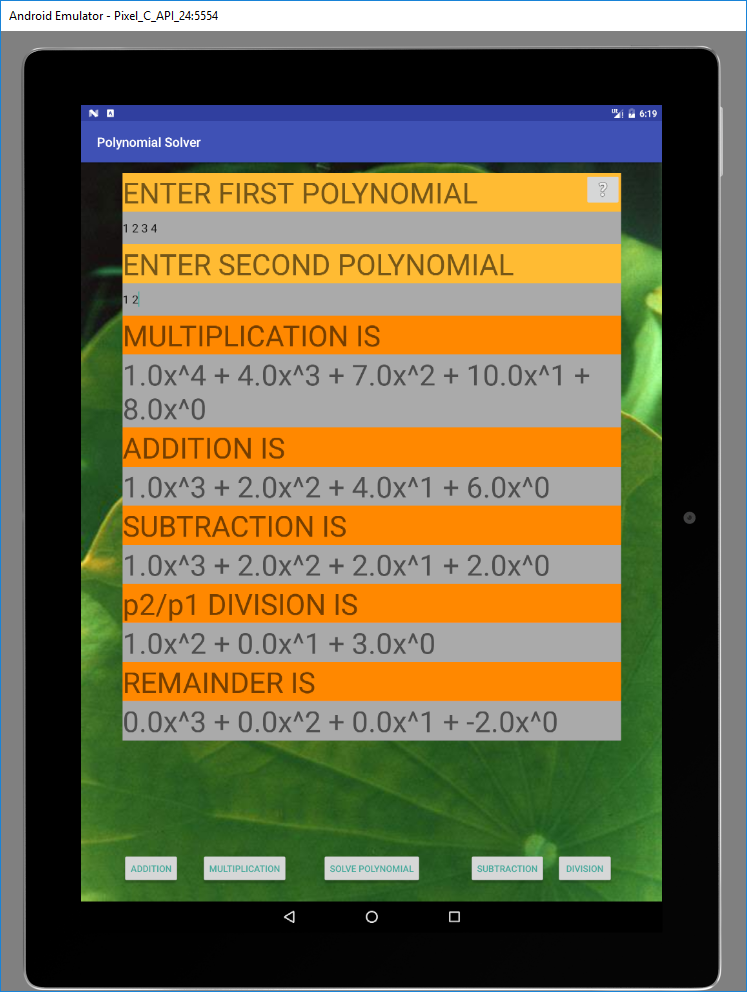
Flow chart

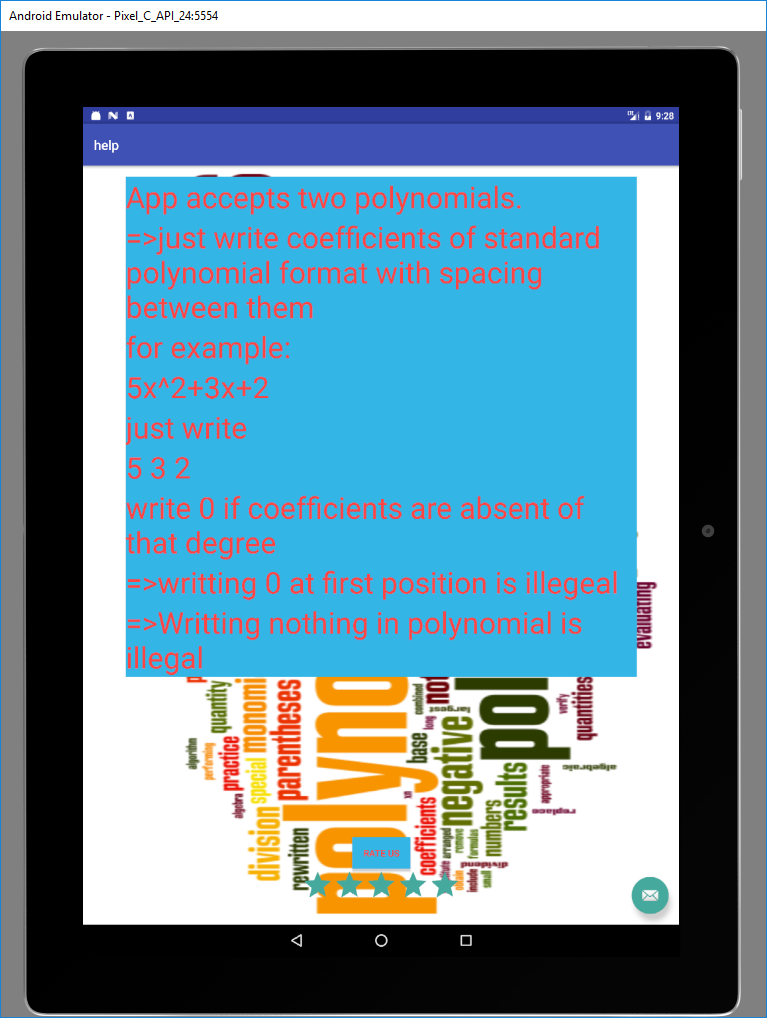


Screenshots

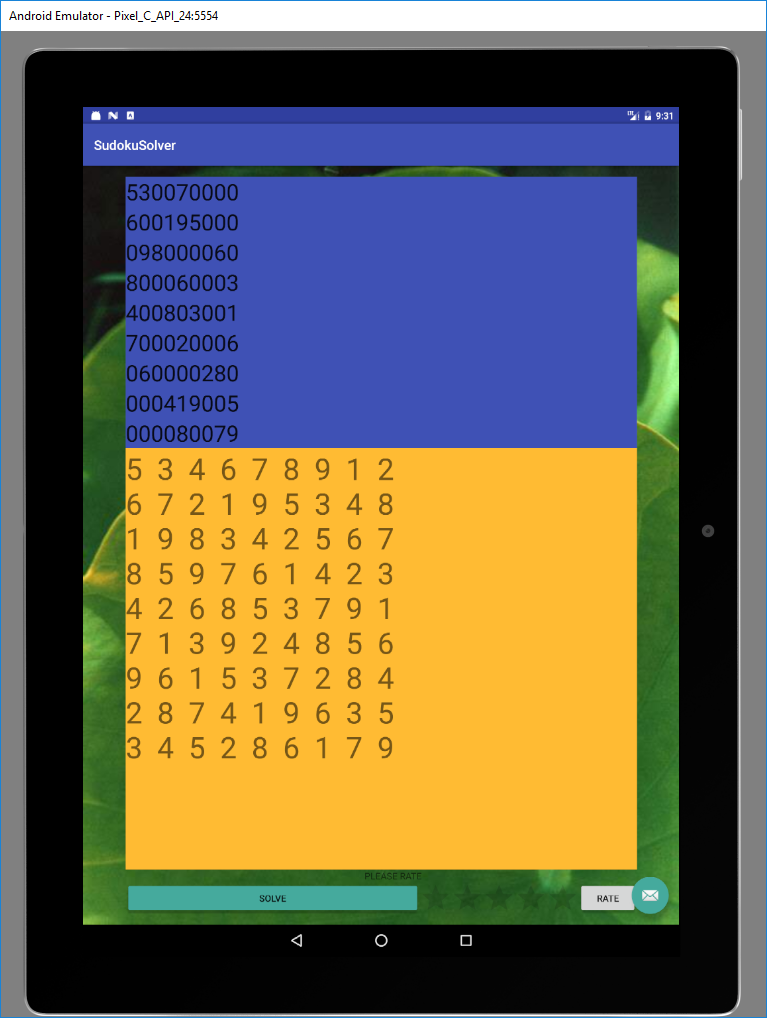
* Screenshots











Limitations of project

* Limitations

1. Can take polynomial of only one variable x.
2. Cannot plot graph
3. Cannot find roots of polynomial.
4. Designed for android tablet pixel c.

Outcome

* Outcome:

I Learned java, Advanced java, xml scripting, Android programming. Project building and thinking capacity increases. Self learning is best choice for project building.Also Learned graphical programming.

Future enhancement

* Future enhancement

1. Graph plotting feature.
2. Root finding feature.
3. Analyze of graph
4. Including mathematical Api.
5. Including graph Api.
6. Actions in new intent layouts.
7. Showing step by step process of solving

References

* **References**

1. https://www.mathsisfun.com/algebra/polynomials.html
2. https://en.wikipedia.org/wiki/Polynomial
3. http://www.softschools.com/math/algebra/topics/simplifying\_polynomials/
4. http://mathworld.wolfram.com/Polynomial.html
5. https://www.youtube.com/watch?v=fGThIRpWEE4

Project Applications

**PROJECT APPLICATIONS**

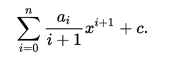
**Calculus**

The simple structure of polynomial functions makes them quite useful in analyzing general functions using polynomial approximations. An important example in calculus is Taylor's theorem, which roughly states that every differentiable function locally looks like a polynomial function, and the [Stone–Weierstrass theorem](https://en.wikipedia.org/wiki/Stone%E2%80%93Weierstrass_theorem), which states that every [continuous function](https://en.wikipedia.org/wiki/Continuous_function) defined on a [compact](https://en.wikipedia.org/wiki/Compact_space) [interval](https://en.wikipedia.org/wiki/Interval_(mathematics)) of the real axis can be approximated on the whole interval as closely as desired by a polynomial function.

Calculating derivatives and integrals of polynomial functions is particularly simple. For the polynomial function{\displaystyle \sum \_{i=0}^{n}a\_{i}x^{i}}the derivative with respect to *x* is

{\displaystyle \sum \_{i=1}^{n}a\_{i}ix^{i-1}}

and the indefinite integral is

{\displaystyle \sum \_{i=0}^{n}{a\_{i} \over i+1}x^{i+1}+c.}

**Abstract algebra**

*Main article:*[*Polynomial ring*](https://en.wikipedia.org/wiki/Polynomial_ring)

In [abstract algebra](https://en.wikipedia.org/wiki/Abstract_algebra), one distinguishes between *polynomials* and *polynomial functions*. A *polynomial* *f* in one indeterminate *x* over a [ring](https://en.wikipedia.org/wiki/Ring_(mathematics)) *R* is defined as a formal expression of the form

{\displaystyle f=a\_{n}x^{n}+a\_{n-1}x^{n-1}+\cdots +a\_{1}x^{1}+a\_{0}x^{0}}

where *n* is a natural number, the coefficients *a*0, . . ., *an* are elements of *R*, and *x* is a formal symbol, whose powers *xi* are just placeholders for the corresponding coefficients *ai*, so that the given formal expression is just a way to encode the sequence (*a*0, *a*1, . . .), where there is an *n* such that *ai* = 0 for all *i* > *n*. Two polynomials sharing the same value of *n* are considered equal if and only if the sequences of their coefficients are equal; furthermore any polynomial is equal to any polynomial with greater value of *n* obtained from it by adding terms in front whose coefficient is zero. These polynomials can be added by simply adding corresponding coefficients (the rule for extending by terms with zero coefficients can be used to make sure such coefficients exist). Thus each polynomial is actually equal to the sum of the terms used in its formal expression, if such a term *aixi* is interpreted as a polynomial that has zero coefficients at all powers of *x* other than *xi*. Then to define multiplication, it suffices by the [distributive law](https://en.wikipedia.org/wiki/Distributive_law) to describe the product of any two such terms, which is given by the rule

{\displaystyle ax^{k}\;bx^{l}=abx^{k+l}}     for all elements *a*, *b* of the ring *R* and all [natural numbers](https://en.wikipedia.org/wiki/Natural_numbers) *k* and *l*.

Thus the set of all polynomials with coefficients in the ring *R* forms itself a ring, the *ring of polynomials* over *R*, which is denoted by *R*[*x*]. The map from *R* to *R*[*x*] sending *r* to *rx*0 is an injective homomorphism of rings, by which *R* is viewed as a subring of *R*[*x*]. If *R* is [commutative](https://en.wikipedia.org/wiki/Commutative_ring), then *R*[*x*] is an [algebra](https://en.wikipedia.org/wiki/Algebra_(ring_theory)) over *R*.

One can think of the ring *R*[*x*] as arising from *R* by adding one new element *x* to *R*, and extending in a minimal way to a ring in which *x* satisfies no other relations than the obligatory ones, plus commutation with all elements of *R* (that is *xr* = *rx*). To do this, one must add all powers of *x* and their linear combinations as well.

Formation of the polynomial ring, together with forming factor rings by factoring out [ideals](https://en.wikipedia.org/wiki/Ideal_(ring_theory)), are important tools for constructing new rings out of known ones. For instance, the ring (in fact field) of complex numbers, which can be constructed from the polynomial ring *R*[*x*]over the real numbers by factoring out the ideal of multiples of the polynomial *x*2 + 1. Another example is the construction of [finite fields](https://en.wikipedia.org/wiki/Finite_field), which proceeds similarly, starting out with the field of integers modulo some [prime number](https://en.wikipedia.org/wiki/Prime_number) as the coefficient ring *R* (see [modular arithmetic](https://en.wikipedia.org/wiki/Modular_arithmetic)).

If *R* is commutative, then one can associate to every polynomial *P* in *R*[*x*], a *polynomial function* *f* with domain and range equal to *R* (more generally one can take domain and range to be the same [unital](https://en.wikipedia.org/wiki/Unital_algebra" \o "Unital algebra) [associative algebra](https://en.wikipedia.org/wiki/Associative_algebra) over *R*). One obtains the value *f*(*r*) by [substitution](https://en.wikipedia.org/wiki/Substitution_(algebra)) of the value *r* for the symbol *x* in *P*. One reason to distinguish between polynomials and polynomial functions is that over some rings different polynomials may give rise to the same polynomial function (see [Fermat's little theorem](https://en.wikipedia.org/wiki/Fermat%27s_little_theorem) for an example where *R* is the integers modulo *p*). This is not the case when *R* is the real or complex numbers, whence the two concepts are not always distinguished in [analysis](https://en.wikipedia.org/wiki/Analysis_(mathematics)). An even more important reason to distinguish between polynomials and polynomial functions is that many operations on polynomials (like [Euclidean division](https://en.wikipedia.org/wiki/Euclidean_division)) require looking at what a polynomial is composed of as an expression rather than evaluating it at some constant value for *x*.

**Divisibility**

In [commutative algebra](https://en.wikipedia.org/wiki/Commutative_algebra), one major focus of study is *divisibility* among polynomials. If *R* is an [integral domain](https://en.wikipedia.org/wiki/Integral_domain) and *f* and *g* are polynomials in *R*[*x*], it is said that *f* *divides* *g* or *f* is a divisor of *g* if there exists a polynomial *q* in *R*[*x*] such that *f* *q* = *g*. One can show that every zero gives rise to a linear divisor, or more formally, if *f* is a polynomial in *R*[*x*] and *r* is an element of *R* such that *f*(*r*) = 0, then the polynomial (*x* − *r*) divides *f*. The converse is also true. The quotient can be computed using the [polynomial long division](https://en.wikipedia.org/wiki/Polynomial_long_division). If *F* is a [field](https://en.wikipedia.org/wiki/Field_(mathematics)) and *f* and *g* are polynomials in *F*[*x*] with *g* ≠ 0, then there exist unique polynomials *q* and *r* in *F*[*x*] with

{\displaystyle f=q\,g+r}and such that the degree of *r* is smaller than the degree of *g* (using the convention that the polynomial 0 has a negative degree). The polynomials *q* and *r* are uniquely determined by *f* and *g*. This is called [*Euclidean division*](https://en.wikipedia.org/wiki/Euclidean_division_of_polynomials)*, division with remainder* or *polynomial long division* and shows that the ring *F*[*x*] is a [Euclidean domain](https://en.wikipedia.org/wiki/Euclidean_domain).

Analogously, *prime polynomials* (more correctly, [*irreducible polynomials*](https://en.wikipedia.org/wiki/Irreducible_polynomial)) can be defined as *non-zero polynomials which cannot be factorized into the product of two non-constant polynomials*. In the case of coefficients in a ring, *"non-constant"* must be replaced by *"non-constant or non-*[*unit*](https://en.wikipedia.org/wiki/Unit_(ring_theory))*"* (both definitions agree in the case of coefficients in a field). Any polynomial may be decomposed into the product of an invertible constant by a product of irreducible polynomials. If the coefficients belong to a field or a [unique factorization domain](https://en.wikipedia.org/wiki/Unique_factorization_domain) this decomposition is unique up to the order of the factors and the multiplication of any non-unit factor by a unit (and division of the unit factor by the same unit). When the coefficients belong to integers, rational numbers or a finite field, there are algorithms to test irreducibility and to compute the factorization into irreducible polynomials (see [Factorization of polynomials](https://en.wikipedia.org/wiki/Factorization_of_polynomials)). These algorithms are not practicable for hand-written computation, but are available in any [computer algebra system](https://en.wikipedia.org/wiki/Computer_algebra_system). [Eisenstein's criterion](https://en.wikipedia.org/wiki/Eisenstein%27s_criterion) can also be used in some cases to determine irreducibility.

**Other applications**

Polynomials serve to approximate other [functions](https://en.wikipedia.org/wiki/Function_(mathematics)" \o "Function (mathematics)).such as the use of [splines](https://en.wikipedia.org/wiki/Spline_(mathematics)" \o "Spline (mathematics)).Polynomials are frequently used to encode information about some other object. The [characteristic polynomial](https://en.wikipedia.org/wiki/Characteristic_polynomial) of a matrix or linear operator contains information about the operator's [eigenvalues](https://en.wikipedia.org/wiki/Eigenvalue). The [minimal polynomial](https://en.wikipedia.org/wiki/Minimal_polynomial_(field_theory)) of an [algebraic element](https://en.wikipedia.org/wiki/Algebraic_element) records the simplest algebraic relation satisfied by that element. The [chromatic polynomial](https://en.wikipedia.org/wiki/Chromatic_polynomial) of a [graph](https://en.wikipedia.org/wiki/Graph_(discrete_mathematics)) counts the number of proper colourings of that graph.The term "polynomial", as an adjective, can also be used for quantities or functions that can be written in polynomial form. For example, in [computational complexity theory](https://en.wikipedia.org/wiki/Computational_complexity_theory) the phrase [*polynomial time*](https://en.wikipedia.org/wiki/Polynomial_time) means that the time it takes to complete an [algorithm](https://en.wikipedia.org/wiki/Algorithm) is bounded by a polynomial function of some variable, such as the size of the input.