**Project Name : Monoalphabetic And Polyalphabetic Cipher**

**Phase 1 : Establish Requirements And Design A**

**Simple Interactive Design**

**Student name : Nikhila Konda**

**Course Number : CS630-740**

**Instructor Name : Dr. Suhair Amer, Ph.D.**

**PROJECT DESCRIPTION**

In this project I am going to design a learning tool for students to learn math. This is the phase 1 of the project which best describes the establish requirements and design a simple interactive design. My topic is about the Monoalphabetic and Polyalphabetic Cipher.

In Cryptography, a substitution cipher is a method of encrypting by which units of plaintext are replaced with ciphertext, according to a fixed system; the "units" may be single letters (the most common), pairs of letters, triplets of letters, mixtures of the above, and so forth. The receiver deciphers the text by performing the inverse substitution. Substitution ciphers can be compared with  transposition ciphers. In a transposition cipher, the units of the plaintext are rearranged in a different and usually quite complex order, but the units themselves are left unchanged. By contrast, in a substitution cipher, the units of the plaintext are retained in the same sequence in the ciphertext, but the units themselves are altered. There are a number of different types of substitution cipher. If the cipher operates on single letters, it is termed a simple substitution cipher; a cipher that operates on larger groups of letters is termed polygraphic. A monoalphabetic cipher uses fixed substitution over the entire message, whereas a polyalphabetic cipher uses a number of substitutions at different positions in the message, where a unit from the plaintext is mapped to one of several possibilities in the ciphertext and vice versa. Monoalphabetic cipher is a substitution cipher in which for a given key, the cipher alphabet for each plain alphabet is fixed throughout the encryption process. For example, if 'A' is encrypted as 'D', for any number of occurrence in that plaintext, 'A' will always get encrypted to 'D'. A polyalphabetic cipher is any cipher based on substitution, using multiple substitution alphabets. The vigenere cipher is probably the best-known example of a polyalphabetic cipher, though it is a simplified special case. The Enigma machine is more complex but is still fundamentally a polyalphabetic substitution cipher. Polyalphabetic ciphers are much stronger. There are many different types, and the strength varies considerably. The clue is in ‘poly’, meaning many. The best known is the Vigenere system (though he didn’t invent it) which uses 26 alphabets and a keyword to tell which alphabet to use for each letter to be encrypted. You can easily apply a progressive key (where keys are used in order one after another, the simplest just advances the letters of the key by one place at each repetition) or an auto key (where the message itself determines which alphabets are to be used for parts of the message after the initial key).

By using this concept description, I am going to develop a Gaming Platform to learn it in an ease way. This Gaming concept is very useful for any level of students to learn the subject. So here I’m using Scratch Programming Language to develop this Gaming tool.

**1. USABILITY AND USER EXPERIENCE GOALS:**

Scratch is a block-based visual programming language and site focused on essentially at youngsters. Clients of the website can make online ventures utilizing a square like interface. The administration is created by the MIT Media Lab, has been converted into 70+ dialects, and is utilized in many pieces of the world. Scratch isn't only a decent prologue to programming; it is a genuine programming language in its own right. Scratch has the important features of many programming languages, like loops (repeat blocks) and conditionals (if-then blocks). In Scratch, improvement is significant for bigger ventures since Scratch is a deciphered language. Proficient programming likewise permits you to have progressively content or a littler document size in a Scratch venture, in light of the fact that the contents would occupy less room, letting you make more use out of 50 MB. Scratch is secure, and ordinarily these individuals are immediately revealed and prohibited. Simply don't give out close to home data and report anyone who is mentioning individual data or in any case making you awkward and you ought to be fine. It merits referencing, however, that you will locate the intermittent grown-up. Scratch has the significant highlights of many programming dialects, similar to circles and conditionals . With these highlights, Scratchers can make calculations, or guidelines to finish explicit undertakings. PC researchers would state Scratch is a Turing-complete programming language, which implies it can play out all the fundamental capacities that make up calculations. In the wake of working with Scratch for some time, numerous Scratchers find that they need to keep utilizing Scratch even as they utilize other programming dialects. Scratch is fine for showing small kids programming ideas. I generally suggest beginning with a decent educating language. A decent training language is exceptionally straightforward and simple to learn. It's liberated from the cruft that you find in dialects like Java, Python, JavaScript and C. Scratch is a simple to-utilize programming stage with the possibility to make stunning intuitive stories, games, films, and introductions. Scratch is a program that permits children to make their own intuitive stories, activities, and games. Scratch is sans fun, and simple to use with extraordinary instructive worth. It's an extraordinary method to figure out how to code in an intuitive manner. The gathering discharged Scratch 3.0 recently. Scratch programming language has a ton of fun, happy play area with a lot of documentation. With every adaptation of Scratch, clients acquire understanding and get happy with controlling their virtual condition. The intuitive programming style is an amazing, low-pressure prologue to the idea of building programs. Learning Scratch includes controlling code hinders inside the play area. The hearty online network offers help and documentation with the goal that scratch clients have a lot of assets. In any case, courses intended to assist you with taking advantage of this programming condition could be an effective method to break into it. You can get into Scratch all alone effectively, however some of the time having bit by bit headings to building help kick off your learning. These programming ideas could make ready for progressively critical tasks not far off with other programming dialects, so beginning is your initial step. Harvey Mudd offers a course in Scratch Programming Basics in organization with edX.org's instructive stage. You'll get familiar with the basics to kick you off, so you invest less energy befuddled and additional time building. In case you're an educator, Delft's course explicitly addresses the aptitudes instructors need to assemble applications in Scratch. In the event that your center is permitting your kids to investigate, Delft's course, Scratch Programming for Kids (8+) is intended to kick kids off in the Scratch play area. Youngsters will require coding abilities for employments later on, so beginning your kids out in grade school could be a superb method to assemble those aptitudes early.

**Desirable Aspects:**

Scratch is intended to be fun, instructive, and simple to learn. It has devices for making intuitive stories, games, craftsmanship, recreations, and that's just the beginning, utilizing square based programming. Scratch additionally has its own paint editorial manager and sound proofreader worked in. Scratch is a slide-and-drop programming language for youngsters that permits them to investigate and try different things with the ideas of PC programming by utilizing the basic graphical interface of "hinders" that are assembled to make basic codes. One of the most intriguing realities of this visual programming language is the inclusion of the network. At the point when somebody completes a task, or at whatever point a particular client needs to, they share and can examine their manifestations with different individuals from the network. network building and activities like Scratch put a great deal of accentuation on this issue, empowering children to grow much more aptitudes and capacities through the learning of coding. Along these lines, activities like Scratch, that were made in a benevolent and vivid manner are an astonishing route for our children to begin creating significant aptitudes, for example, computational reasoning, algorithmic rationale, critical thinking and inventiveness. The advantages don't stop there, there's substantially more to include. By being a piece of a network of producers and makers, our children will have the option to get the entirety of the criticism they have to fortify their turns of events, to tune in to other people who may have confronted comparative issues previously and to buckle down on another significant capacity for the 21st Century. You'd be astounded to figure out the fact that it is so natural to program by utilizing Scratch's squares. With a very easy to use interface, and with appealing hues, Scratch's originators made the perfect stage to start the way toward figuring out how to code. At that point, going to a programming language like Java, C++, or even Python will be a way progressively regular procedure for our darling little students.

**Undesirable Aspects:**

Scratch is focused at 8-multi year old yet the site has highlights like sprites in the editorial manager that could be viewed as whimsical, just as the site concealing about all activities with an extremely modest quantity of viciousness in them, despite the fact that they are less brutal than E10+ or 7+ games. A great deal of activities that persuade highlighted appear to be ventures that are progressively open to more youthful youngsters, as opposed to ventures that exhibit the assortment of things that can be made utilizing Scratch's customizing. As opposed to thinking about Scratch as a "kids' modifying language", it ought to be advanced as a visual programming language simple for fledglings to utilize that is open and reasonable for youngsters also. 8-multi year olds aren't little youngsters, however there is by all accounts a conviction among non-Scratch software engineers that Scratch is a kids' "toy" programming language, when it's definitely not. It is acceptable to not have this impression engendered further. The Tips page contains instructional exercises for straightforward undertakings, so it doesn't make it quickly evident to new individuals that Scratch accomplishes something beyond vivify your name and permit you to make a Pong game. It would be useful for Scratch to cause more to notice how these ventures are customized, just as show instances of increasingly complex activities to represent what clients can make once they've learnt more aptitudes in programming. This is more valid for games than different kinds of tasks, yet frequently exceptionally straightforward games get included, and complex games regularly aren't. Seeing as the first page is the main thing individuals see, it doesn't do a lot to show the assortment of undertakings that can be made. Progressively intricate activities ought to be included also - you can't expect they will get well known all alone.

**2. QUESTIONS:**

General Questions:

1. What is the purpose of scratch?
2. What License are scratch programs under?
3. Can I sell my scratch projects?

Community Guidelines Questions:

1. [How do I report inappropriate content?](https://en.scratch-wiki.info/wiki/Frequently_Asked_Questions#How_do_I_report_inappropriate_content.3F)
2. [What do I do if I see a Scratcher violating the Community Guidelines?](https://en.scratch-wiki.info/wiki/Frequently_Asked_Questions#What_do_I_do_if_I_see_a_Scratcher_violating_the_Community_Guidelines.3F)
3. [Why are some users blocked from accessing the Scratch Website?](https://en.scratch-wiki.info/wiki/Frequently_Asked_Questions#Why_are_some_users_blocked_from_accessing_the_Scratch_Website.3F)

Scratch program Questions:

1. [What are the system requirements for Scratch?](https://en.scratch-wiki.info/wiki/Frequently_Asked_Questions#What_are_the_system_requirements_for_Scratch.3F)
2. [Can Scratch's project resolution be changed?](https://en.scratch-wiki.info/wiki/Frequently_Asked_Questions#Can_Scratch.27s_project_resolution_be_changed.3F)
3. [Why don't my projects work correctly online?](https://en.scratch-wiki.info/wiki/Frequently_Asked_Questions#Why_don.27t_my_projects_work_correctly_online.3F)

Programming Questions:

1. How do I stop variables from resetting to their default positions?
2. Why does my sprite appear upside-down when rotating?
3. Why can't I save my costumes or sounds?

Website Questions:

1. Why do new users have to wait between comments?
2. Will Scratch ever get private messaging?
3. How do I get my projects to become popular?

Account Questions:

1. How can I reset my password?
2. I forgot my username. How can I get it back?
3. How do I change my user icon?

Forum Questions

1. How do I become a Scratcher?
2. What is the difference between a Scratcher and a New Scratcher?
3. How do I create a signature?

Scratch Modification Questions

1. What license is the Scratch source code under?
2. What is a Scratch Modification?

Reference Questions

1. What is the point of Scratch modifications?
2. How can I create my own Scratch modification?

**3. User Need’s and User Requirements And Main Tasks:**

As of March 2020, there are more than 50 million clients. To discover state-of-the-art and increasingly precise and accurate data, visit the insights page situated at the base of the page on the footer, where you can discover more measurements, for example, the complete number of undertakings, studios and remarks on Scratch. Numerous clients learn Scratch as they go, attempting orders from the palette or investigating code from existing undertakings. To energize such self-coordinated learning, the Scratch programming condition was intended to welcome scripting, give prompt input to content execution, and make execution and information obvious. A key objective of Scratch is to acquaint programming with those with no past programming experience. This objective drove numerous parts of the Scratch plan. A portion of the plan choices are self-evident, for example, the decision of a visual square’s language, the single-window UI design, and the insignificant order set. Others are more subtle, for example, how the intended interest group affected the type framework and the way to deal with mistake taking care of. We state that Scratch is tinkerable in light of the fact that it lets clients explore different avenues regarding orders and code bits the manner in which one may tinker with mechanical or electronic segments. Tinkerability empowers hands-on learning and supports a base up way to deal with composing contents where little pieces of code are amassed and tried, at that point consolidated into bigger units. Tinkerability assists clients with finding the usefulness of squares. A square can be tried by tapping on it, even in the palette. Capacity squares show their return an incentive in an animation like "talk bubble".

To help clients all the more effectively investigate what squares do, each square accompanies default parameters that give a lighting up exhibition of what that square does. Scratch has help screens for each order, available through the right-button menu, however numerous clients find out about orders just by attempting them. Scratch doesn't necessitate that the client make total contents before running the undertakings. Program parts can be left in the scripting sheet and are spared with the undertaking. Scratch gives visual input to show content execution. At the point when a content is running, it is encompassed by a gleaming white fringe. This input enables the client to comprehend when contents are activated and to what extent they run. On the off chance that a content experiences a mistake the fringe turns red and the square that caused the blunder is featured in red. Scratch can likewise show order sequencing and stream of control. Empowering single-venturing makes squares streak as they run. In any event, when single-venturing mode isn't empowered, Scratch refreshes the showcase after each order. Seeing the impact of each order, regardless of whether just as a concise glimmer on the screen, gives significant visual pieces of information while investigating. Little plan subtleties can have a major effect. In early forms of Scratch, a recently made variable was not shown on the stage, and the signal to get that going was not self-evident. Numerous clients didn't find factors, in any event, when they required them. Subsequent to changing the plan so a recently made variable is promptly shown, a lot more clients began utilizing factors. The Scratch programming framework endeavors to assist clients with building instincts about PC programming as they make extends that connect with their inclinations. The UI format, with its unmistakable order palette and focal scripting region, welcomes clients to program. The Scratch squares language disposes of grammar blunders, permitting clients to concentrate on intriguing issues immediately, rather than battling only to get their program to aggregate. Square shapes and visual criticism while hauling help the client figure out how to collect projects also, use information types.

The Main tasks involved in this scratch programming are :

a) Data Collection

b) Calculation of Programming Mastery Scores

c) Importing the Data

d) Data Representation

e) Data set Contents

f) Data set Extension

Various works in the registering training research field endeavor to survey the programming abilities that fledgling software engineers create in the Scratch condition. While Scratch is getting expanding enthusiasm as a starting programming language, there is no ongoing informational index of Scratch programs accessible to the examination network. To gather the information from the web interface of the Scratch venture storehouse, we manufactured a scratching program. The web scratching program begins by perusing the Scratch ventures and in this manner acquires the task identiﬁers of activities that were most as of late shared. Along these lines, it recovers a JSON ﬁle for every one of the recorded undertakings. source code to appoint scores on seven components of computational reasoning: deliberation and issue decay, intelligent reasoning, synchronization, parallelism, algorithmic thoughts of ﬂow control, client intelligence and information portrayal. All scratched venture information and metadata, including the rundown of utilized squares and parameters, were imported in a social database. We likewise imported the information on the shapes and the classes of the Scratch squares. We at that point utilized SQL questions for normalizing the information and getting it its ﬁnal construction. The activities' information are put away in a social database. Every one of the undertakings is identiﬁed by its Scratch venture ID,stored in ﬁeld p\_id, while its creator is identiﬁed by the username. On the off chance that a venture is a remix of another, the first undertaking can be found in the remixes table. Table evaluations stores the Dr. Scratch results for the programming dominance measurements per venture. The diagram of the database, reﬂects the structure of the Scratch programs. Tasks contain sprites, which are elements with their own related code. The code is composed into contents, i.e., gatherings of Scratch code blocks, each content having a place with a sprite named sprite-name. For instance, there are 3 contents, one started when the green ﬂag is clicked, one then the sprite is clicked, and the 'back ﬂip' custom square deﬁnition. Those custom squares are what might be compared to technique deﬁnitions, with their names and contentions put away in the systems table. Most of the undertakings have been seen just once, and they have not been favorited or remixed. Around 10,000 of the absolute activities have been remixed in any event once, while there are instances of extremely well-known ventures with more than 100 remixes and even one thousand cases within excess of 100 perspectives. The assessment of square based dialects when all is said in done, and Scratch specifically, as instruments for programming instruction is getting signiﬁcant inquire about consideration. Various examinations have been completed during the most recent years on understanding the programming practices of students in those environments, on the programming aptitudes they create, and on the nature of their projects. We introduced an informational collection of ongoing Scratch programs scratched from the Scratch venture vault. It is made accessible as a database which incorporates the source code of the Scratch extends, their metadata, and their programming dominance scoring results.

**4. Scenarios And Use Cases:**

Computational Thinking Design Scenarios – Part of a Computational Thinking Assessment Strategy.

These scenarios were created as a team with scientists at Education Development Center (EDC) as a feature of an NSF award concentrated on the improvement of computational thoroughly considering Scratch programming exercises. We created three arrangements of Scratch ventures with expanding multifaceted nature. Inside each set, there were two ventures; the activities connected with similar ideas and practices yet had various feel to speak to various interests. In a progression of three meetings, understudies were given the plan situations, which were confined as tasks that were made by another youthful Scratcher. The understudies were then solicited to choose one from the tasks from each set, and

a) clarify what they chose venture does,

b) portray how it could be expanded,

c) fix a bug, and

d) remix the undertaking by including an element.

The mix of these four exercises rose up out of a few autonomous exercises (introductions, scrutinizes, investigating, challenges, and remixing) that we had been exploring different avenues regarding in workshops for youthful Scratchers and teachers.

A monoalphabetic figure is one where every image in the information (known as the ''plaintext'' is mapped to a fixed image in the yield (alluded to ciphertext). Polyalphabetic figure is any figure dependent on substitution, utilizing various substitution letter sets. In monoalphabetic Cipher, when a key is picked, each alphabetic character of plaintext is mapped onto an extraordinary alphabetic character of a ciphertext. Then again, in polyalphabetic figure, each alphabetic character of plaintext can be mapped onto ''m'' alphabetic characters of a ciphertext. In monoalphabetic Cipher, the connection between a character in the plaintext and the characters in the ciphertext is balanced, while in Polyalphabetic Cipher, the connection between a character in the plaintext and the characters in the ciphertext is one-to-many. Monoalphabetic Cipher incorporates added substance, multiplicative, relative and monoalphabetic substitution figure. Then again, Polyalphabetic figure incorporates Autokey, Playfair, Roto, One-time cushion, Enigma figure and Vigenere. For a stream figure to be a monoalphabetic figure, the estimation of Ki doesn't rely upon the situation of the plaintext character in the plaintext stream. Then again, for a stream to be a polyalphabetic figure, the estimation of id doesn't rely upon the situation of the plaintext stream. A basic substitution figure includes a solitary mapping of the plaintext letters in order onto figure content characters. A more complex option is to utilize diverse substitution mappings (called numerous letters in order) on different parts of the plaint ext. This outcomes in supposed polyalphabetic substitution. ith transposition and some other complex changes to make a figure content from a plain book. In our paper we put accentuation on proposing another mix technique, Vigenere figure with Affine figure, on the grounds that the figure dependent on basic Vigenere strategy isn't secure. An Affine figure is a mix of the added substance figure and multiplicative figure. Added substance figure, multiplicative figure and Affine figure are monoalphabetic figure strategies.

**5. Requirements using Volere Shell:**

**a) Project Drivers:**

The purpose of this project is to study about ciphers concept in cryptography which is Monoalphabetic Cipher and Polyalphabetic Cipher and to develop a Learning Game for students. The main moto of this project is to understand the concepts in a simple and easy way so, that everyone learn in small span of time. And it will remain in their brain for the long time. This project is helpful mainly for the students who study High school and in university to the under graduates. Learning with the help of gaming tool is not a bad idea and it also enhances the creativity skills in students. It is easy way to learn and remember the concepts in the brains of students and perform well in exams as well.

**b) Project Constraints:**

A monoalphabetic substitution figure, otherwise called a straightforward substitution figure, depends on a fixed substitution structure. That is, the substitution is fixed for each letter of the letters in order. Along these lines, on the off chance that "an" is scrambled to "R", at that point each time we see the letter "an" in the plaintext, we supplant it with the letter "R" in the ciphertext. The historical backdrop of straightforward substitution figures can be followed back to the soonest civilizations, and for quite a while they were more than satisfactory for the reasons for which they were required. By the present measures they are exceptionally powerless, and unfathomably simple to break, however they were a significant advance in creating cryptography. The advancement of Polyalphabetic Substitution Ciphers was the cryptographers answer to Frequency Analysis. The principal known polyalphabetic figure was the Alberti Cipher created by Leon Battista Alberti in around 1467. He utilized a blended letters in order to scramble the plaintext, yet at irregular focuses he would change to an alternate blended letters in order, demonstrating the change with a capitalized letter in the ciphertext. So as to use this figure, Alberti utilized a figure plate to show how plaintext letters are identified with ciphertext letters. For instance, when the plate on the left is set as appeared, we see that the plaintext letter "e" (outwardly ring) is scrambled to "Z" (within ring). Alberti would utilize this setting for a couple of letters of the message, and afterward turn the inward circle to an alternate setting for the following hardly any letters, etc.

**c) Functional Requirements:**

Cryptography before the cutting-edge age was viably synonymous with encryption, the transformation of data from a comprehensible state to clear rubbish. The originator of an encoded message shares the interpreting system just with proposed beneficiaries to block access from foes. The cryptography writing regularly utilizes the names Alice ("A") for the sender, Bob ("B") for the expected beneficiary, and Eve ("meddler") for the adversary.[5] Since the improvement of rotor figure machines in World War I and the coming of PCs in World War II, the techniques used to do cryptology have gotten progressively mind boggling and its application more widespread. Modern cryptography is vigorously founded on scientific hypothesis and software engineering practice; cryptographic calculations are structured around computational hardness suppositions, making such calculations difficult to break by and by any enemy. It is hypothetically conceivable to break such a framework; however, it is infeasible to do as such by any known handy methods. These plans are consequently named computationally secure; hypothetical advances, e.g., enhancements in whole number factorization calculations, and quicker registering innovation require these answers for be consistently adjusted. There exist data hypothetically secure plans that provably can't be made back the initial investment with boundless figuring power a model is the one-time cushion however these plans are harder to use by and by than the best hypothetically delicate yet computationally secure systems. The development of cryptographic innovation has raised various lawful issues in the data age. Cryptography's potential for use as an apparatus for secret activities and dissidence has driven numerous administrations to group it as a weapon and to constrain or even deny its utilization and fare. In certain wards where the utilization of cryptography is legitimate, laws license specialists to force the exposure of encryption keys for archives pertinent to an examination. Cryptography likewise assumes a significant job in computerized rights the executives and copyright encroachment of advanced media. Distinctive physical gadgets and helps have been utilized to help with figures. One of the most punctual may have been the scytale of old Greece, a pole as far as anyone knows utilized by the Spartans as a guide for a transposition figure. In medieval occasions, different guides were developed, for example, the figure grille, which was likewise utilized for a sort of steganography.

**d) Non-Functional Requirements:**

The objective of cryptanalysis is to discover some shortcoming or uncertainty in a cryptographic plan, accordingly, allowing its disruption or avoidance. It is a typical confusion that each encryption technique can be broken. Regarding his WWII work at Bell Labs, Claude Shannon demonstrated that the one-time cushion figure is unbreakable, given the key material is really arbitrary, never reused, stayed discreet from every conceivable aggressor, and of equivalent or more prominent length than the message. Cryptanalysis of symmetric-key figures normally includes searching for assaults against the square figures or stream figures that are more proficient than any assault that could be against an ideal figure. For instance, a straightforward beast power assault against DES requires one known plaintext and 255 decoding, attempting roughly 50% of the potential keys, to arrive at a point where chances are better than even that the key looked for will have been found. While unadulterated cryptanalysis utilizes shortcomings in the calculations themselves, different assaults on cryptosystems depend on real utilization of the calculations in genuine gadgets and are called side-channel assaults. On the off chance that a cryptanalyst approaches, for instance, the measure of time the gadget took to scramble various plaintexts or report a mistake in a secret phrase or PIN character, he might have the option to utilize a planning assault to break a figure that is in any case impervious to examination.

**e) Project Issues:**

There are no open issues for performing this project using the Scratch Programming. Problems with this learning tool are very less because it is based on Block based programming language and it is very easy to learn and code the project. If at all there arises some problems it is very easy to fix them within no time. And the accessibility is convenient for any group of children for creating projects. We can create multiple number of tasks depending upon our usability. The difficulty level of the task depends upon the level of students learning the topics. When doing the project in this Scratch we can convert them into any other programming language depending upon our purpose of using it. There is not bit of risk involved in this project creation and if arises with small issue scan be fixed in anytime. And it is open source project we can download free and get an access in no time. And can easily develop a project depending upon the difficulty level of children. Cryptography has for quite some time been important to insight social affair and law requirement organizations. Mystery interchanges might be criminal or even treasonous.

**6. Conceptual Model:**

By using this Monoalphabetic and polyalphabetic ciphers concept we can program by using a scratch programming. Programming just alludes to the craft of composing guidelines (calculations) to instruct a PC. Scratch is a visual programming language that gives a perfect learning condition to doing this. Initially created by America's Massachusetts Institute of Technology, Scratch is a basic, visual programming language. Shading coded squares of code essentially snap together in specific manners like a jigsaw, dispensing with the composing mistakes that will in general happen when individuals use content-based programming dialects. Numerous media rich projects can be made utilizing Scratch, including games, activities and intelligent stories. Scratch is very likely the most broadly utilized programming for instructing programming to Key Stage 2 and Key Stage 3. The Scratch interface is divided into two sections: the project running environment and the project development. In the top-right of Scratch is the stage shown at the top of the image on the right. The stage is where a Scratch project is physically run, so when one plays a game, the Stage is the window in which it is run. By default, the Scratch Cat is on the stage. The Scratch Cat is simply one of many Sprites, or characters, buttons, etc. in a project. Characters are programmed to perform what a Scratcher desires them to do. The flexibility of Scratch allows the creator to be imaginative and actually make the desired project. That is when programming comes into place, as it "makes things do what they should".

Before getting more into the interface, the quickest way to understand how sprites are programmed in Scratch is by testing things out. Follow the steps below when the Scratch program is opened with a clean, new project. This best describes every inch of the project to make it easier. Because in this Scratch programming the program of the project is done using blocks in which we have nine types of blocks to create a pattern. Blocks are puzzle-piece shapes that are used to create code in [Scratch](https://en.scratch-wiki.info/wiki/Scratch). The blocks connect to each other vertically like a jigsaw puzzle, where each data type ([hat](https://en.scratch-wiki.info/wiki/Hat), [stack](https://en.scratch-wiki.info/wiki/Stack), [reporter](https://en.scratch-wiki.info/wiki/Reporter), [Boolean](https://en.scratch-wiki.info/wiki/Boolean), or [cap](https://en.scratch-wiki.info/wiki/Cap)) has its own shape and a specially shaped slot for it to be inserted into, which prevents syntax errors. Series of connected blocks are called [scripts](https://en.scratch-wiki.info/wiki/Script). The list blocks are shown under the Variables Blocks. In total, there are seven [Hat Blocks](https://en.scratch-wiki.info/wiki/Hat_Block), five [C Blocks](https://en.scratch-wiki.info/wiki/C_Block), thirty-one [Reporter Blocks](https://en.scratch-wiki.info/wiki/Reporter_Block), thirteen [Boolean Blocks](https://en.scratch-wiki.info/wiki/Boolean_Block), two [Cap Blocks](https://en.scratch-wiki.info/wiki/Cap_Block) and fifty-nine [Stack Blocks](https://en.scratch-wiki.info/wiki/Stack_Block). The Paint Editor is similar to other painting programs, such as Microsoft Paint or Paintbrush. To draw a new sprite, click the Paintbrush button next to New sprite. You can change how sprites look by switching to one of many costumes. To create a new costume for a sprite, click the Costumes tab, and then click the Paintbrush button next to New Costume.

The main parts of the Paint Editor are

a) The drawing tools, which you can select using the buttons on the left side

b) The Canvas, where you draw images

c)  The Costume center, which indicates the center of the costume with the crosshairs symbol

d) The Line width slider, which sets the width of the drawing tools

e) The Color selector, which changes the color of the drawing tools

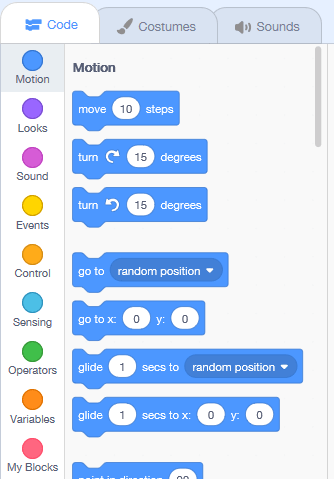
f)The Zoom buttons for zooming into or out of the canvas

g) The Undo and Redo buttons, which can help you correct mistakes.

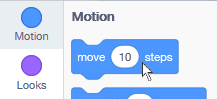
**7. Mental Model:**

Before getting more into the interface, the quickest way to understand how sprites are programmed in Scratch is by testing things out. Follow the steps below when the Scratch program is opened with a clean, new project.

**1.** Access this area of the Scratch program:

[](https://en.scratch-wiki.info/wiki/File:3.0_Block_Selection_Pane.png)

**2.** Select the blue "[block](https://en.scratch-wiki.info/wiki/Blocks)" called [move () steps](https://en.scratch-wiki.info/wiki/Move_()_Steps_(block)), and drag it to the right.

[](https://en.scratch-wiki.info/wiki/File:Grab_Block.png)

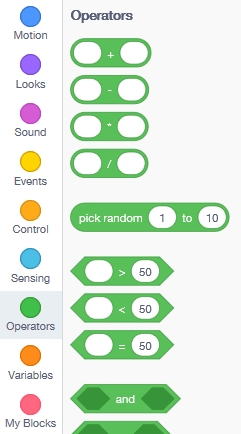
**3.** Release the mouse to place the block; make sure the block is placed in the darker grey, technically called the [scripts area](https://en.scratch-wiki.info/wiki/Scripts_Area).

[](https://en.scratch-wiki.info/wiki/File:Place_Block.png)

**4.** When done, click anywhere on the block except the white middle, and watch what happens to the Scratch Cat... it moves 10 steps.

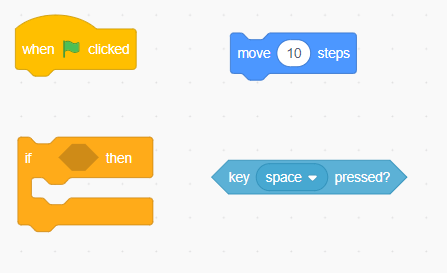
[](https://en.scratch-wiki.info/wiki/File:Scratch_Cat_Move.png)

**5.** Check out the other block categories and test out what each one does!

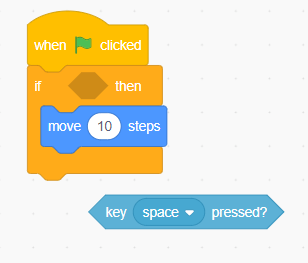
[](https://en.scratch-wiki.info/wiki/File:Category_Select.png)

**Blocks**

As shown above, blocks are the building "blocks" of a Scratch project. They have specific commands which function uniquely from one another. Some blocks can even fit inside *other* blocks, as shown below:  
**1.** Assemble the following "script", or connection of blocks, by accessing the various blocks by color and category.

[](https://en.scratch-wiki.info/wiki/File:Scattered_Blocks.png)

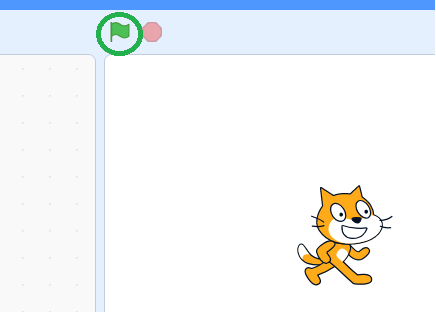
**2.** Assemble the blocks into this formation:

[](https://en.scratch-wiki.info/wiki/File:Assemble_Blocks.png)

**3.** Grab the blue key sensing block that is still in the void and place it into the hexagonal input area of the orange "if" block:

[](https://en.scratch-wiki.info/wiki/File:Block_Input.png)

**4.** Click the [Green Flag](https://en.scratch-wiki.info/wiki/Green_Flag) to run the project, and see what it does!

[](https://en.scratch-wiki.info/wiki/File:Click_Flag.png)

**5.** Unless you were holding down the space key, nothing should have happened. Why is that? Take a look at the script again; remember, a script is a fully connected chain/stack of blocks.

whenclickedifkeyspacepressed?thenmove10steps

The script begins with "when green flag clicked", which was done. When the green flag is clicked, it triggers the script beginning with the "when green flag clicked" block to run. When the script ran, it first detects *if* the space key is down, and *if* it is, *then* the sprite will move 10 steps. Run the project again while holding the space key down, and the sprite will move 10 steps.

**8. Analyze and Enhanced Conceptual Model:**

An away from of Cryptology is fundamental to the investigation of PC security. This is the motivation behind why most early on courses invest a lot of energy talking about traditional figures. These cryptographic frameworks can show numerous ideas and standards. Additionally, the cryptograms they create can be sufficiently basic to be comprehended generally quickly, which offers the understudies a chance to acquaint themselves with diﬀerent sorts of cryptanalytic assaults and procedures, just as comprehend Kerckhoﬀs' proverbs, especially why "just a cryptanalyst can pass judgment on the security of a crypto framework". Thusly we give a strong foundation to the investigation of present day symmetric crypto frameworks. Be that as it may, as simple to fathom as the cryptograms acquired utilizing old style figures might be, gathering the data required for the cryptanalysis. The table contains 26 letter sets written in various columns every letter set being consistently moved to one side as per the past letters in order, comparable to the 26 potential Caesar Ciphers. The figure utilizes an alternate letters in order from one of the lines at different focuses in the encryption procedure. The principal figure gadget seems to have been utilized by the antiquated Greeks around 400 bc for mystery interchanges between military leaders. This gadget, called the scytale, comprised of a decreased twirly doo around which was spirally wrapped a bit of material recorded with the message. In cryptography, a figure is a calculation for performing encryption or decoding a progression of all-around characterized steps that can be followed as a strategy. Another option, less regular term is encipherment. When utilizing a figure, the first data is known as plaintext, and the scrambled structure as ciphertext. Subsequently, before a figure can work, both the sender and beneficiary must have a key or set of keys. Cryptographic figures are utilized to change over ciphertext to plaintext and back. Both of the keys can be utilized to encode a message; the contrary key from the one used to scramble the message is utilized for unscrambling. The contrast between a figure and a code is: a figure changes a message on a letter-by-letter premise, while a code changes over entire plaintext words or expressions into different words or numbers. The Rossini’s' Great Cipher was utilized by Louis XIV of France during the 1600s and was a case of a nomenclator. Figure content is written in scrambled structure and individuals can't comprehend or decipher it while plain content is simply ordinary content written in any human reasonable language like English. Plaintext is a contribution for an encryption calculation and can be in type of content, sound, video and furthermore pictures. A figure is a mystery code, typically one that is made utilizing a scientific calculation. Now and again the message written in code is itself called a figure. From its "zero" which means, figure can likewise be utilized for an individual who has no impact or significance on the planet. Ciphertext is otherwise called scrambled or encoded data since it contains a type of the first plaintext that is indistinguishable by a human or PC without the best possible figure to unscramble it. Decoding, the opposite of encryption, is the way toward transforming ciphertext into discernible plaintext. Monoalphabetic figure is a substitution figure in which for a given key, the figure letters in order for each plain letter set is fixed all through the encryption procedure. For instance, if 'An' is encoded as 'D', for any number of event in that plaintext, 'A' will consistently get scrambled to 'D'. The entirety of the substitution figures we have talked about before in this section are monoalphabetic; these figures are profoundly helpless to cryptanalysis. Polyalphabetic Cipher is a substitution figure in which the figure letter set for the plain letters in order might be distinctive at better places during the encryption procedure. The following two models, Playfair and Vigenere Cipher are polyalphabetic figures.

**9. Interface Design Issues:**

UI structure, or UI configuration, is the specialty of making the interface clients of a site, cell phone application, or other stage connect with. UI plan, without anyone else, is simply models, only demos of what the genuine structure can be. Obviously, UI is urgent to any site or application. UX represents client experience. UX is the partner of UI. UX is the means by which the clients' experience a site or application. It's not the genuine plan of the site, it's the way those components are utilized to make the progression of a site, for example, page route. This is a somewhat confused, and even dubious theme. In any case, let us make one point understood: WE own this studio. Hence, WE can choose what goes in this studio. Presently, kindly don't misinterpret this. We don't abhor you and your work. The truth of the matter is, we can't acknowledge everything that we get to this studio. Along these lines, some of the time your task doesn't arrive at our Design Standards.

Structure GUIDELINES:

Here is our rundown of capabilities for adding a UI idea to this studio.

a) The UI configuration MUST be an idea, implying that is anything but a completely working venture. This additionally applies to ventures that are made to not flaunt a UI plan.

b) The UI configuration ought to utilize a cutting-edge style. Level illustrations, insignificant shading plans, splendid hues, and productive space utilization. We will acknowledge plans that don't follow these structure rules 100% superbly if the plan is especially all around planned regarding design and clearness.

c) The UI configuration MUST NOT be legitimately replicated from some different Scratchers present or past works. It MUST be a unique structure or an elegant overhaul of a current work.

The Scratch graphical user interface (GUI) is divided into four main panes.

Scratch Interface (GUI)

1. Blocks Palette

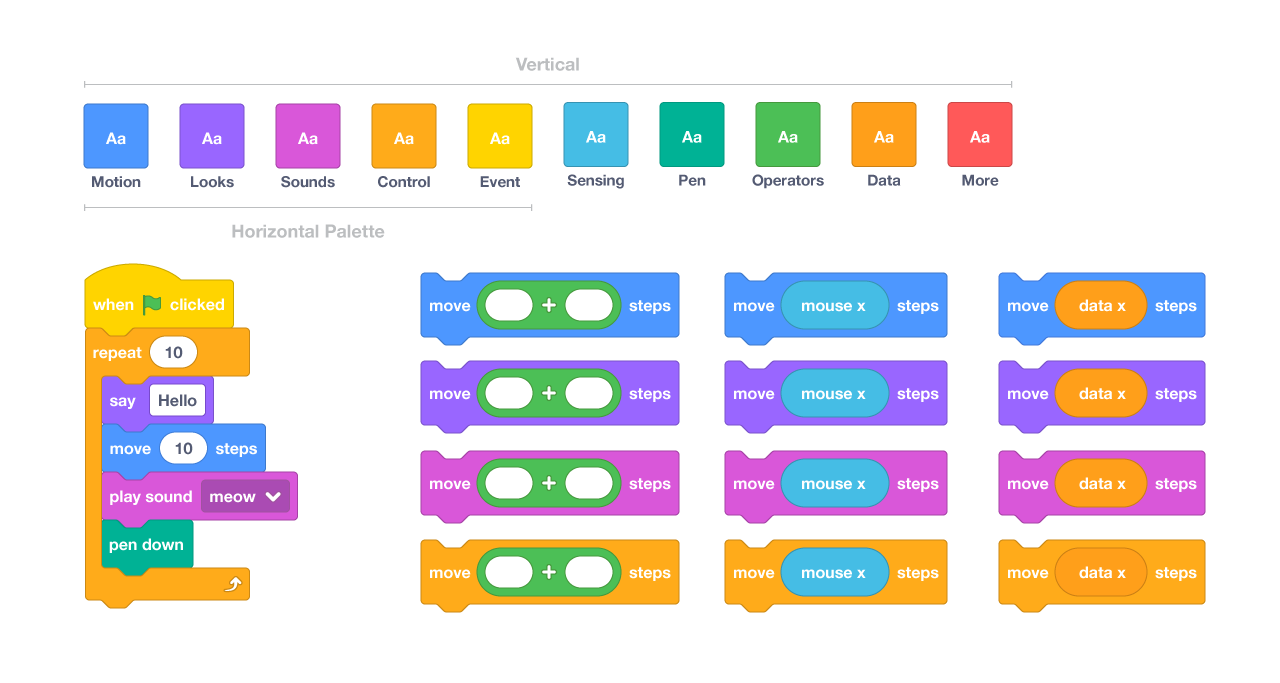
2. Script Area

3. Sprite Area

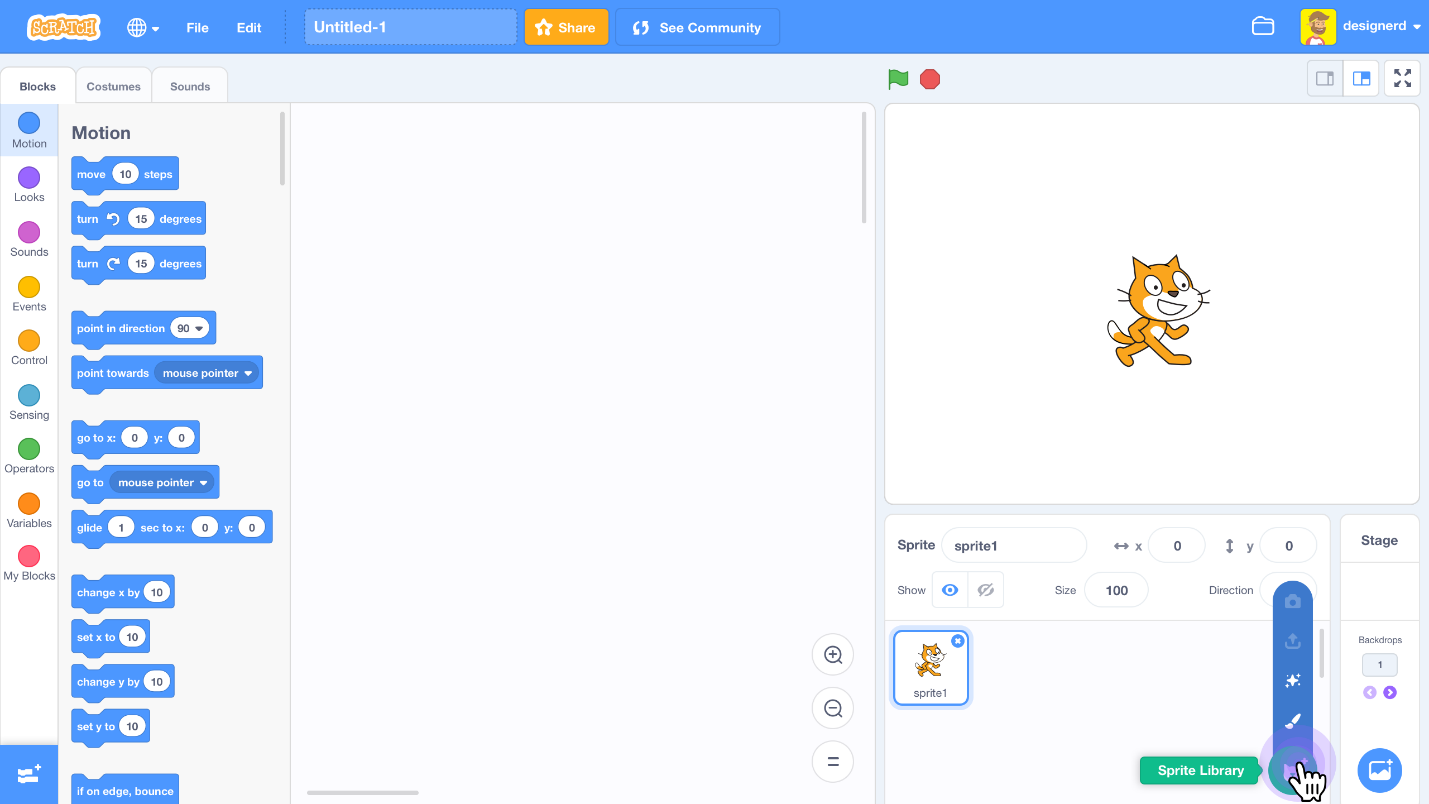
4. Stage

Blocks Palette (left): This area provides a list of commands that includes blocks of conditions, parameters, and consequences. The palettes are organized into eight color-coded groups of blocks. Script Area (center): Commands from the blocks palette may be dragged and dropped to this area so that multiple sequences of commands may be executed for each sprite (character or object) and stage (background). This area also allows the programmer to add or modify costumes (for animation) and sounds. Each sprite and stage has its own script area, costumes, and sounds depending on the sprite or stage selected. Sprite Area (bottom right): This area allows the programmer to select a different sprite or stage, in addition to creating new sprites and importing existing sprites. Stage: This area is where the scripts for the sprites and stages are visually executed.

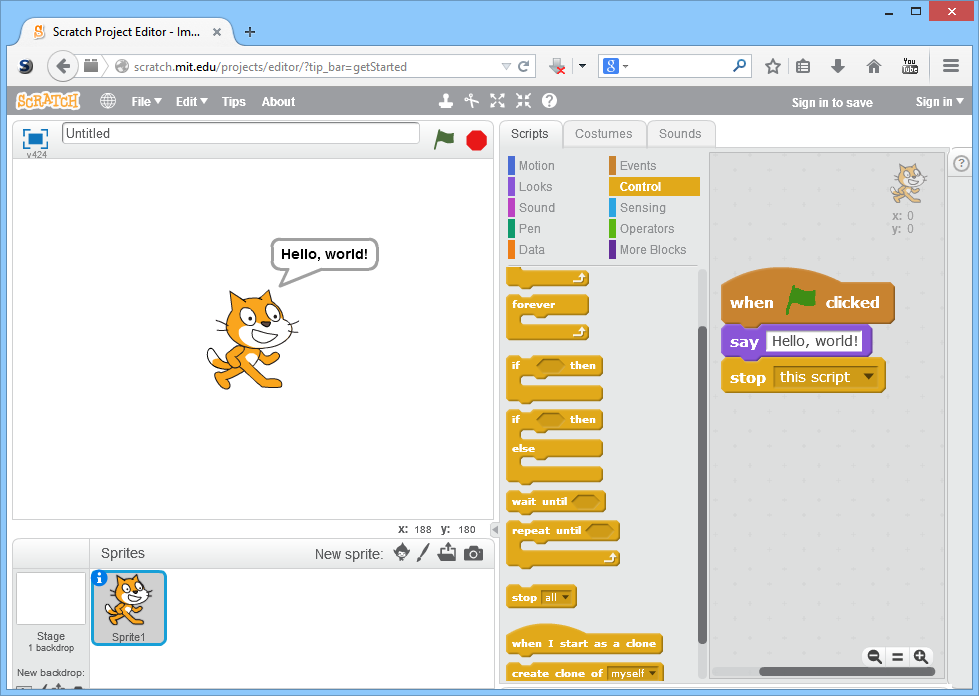
**10. Initial Designs:**



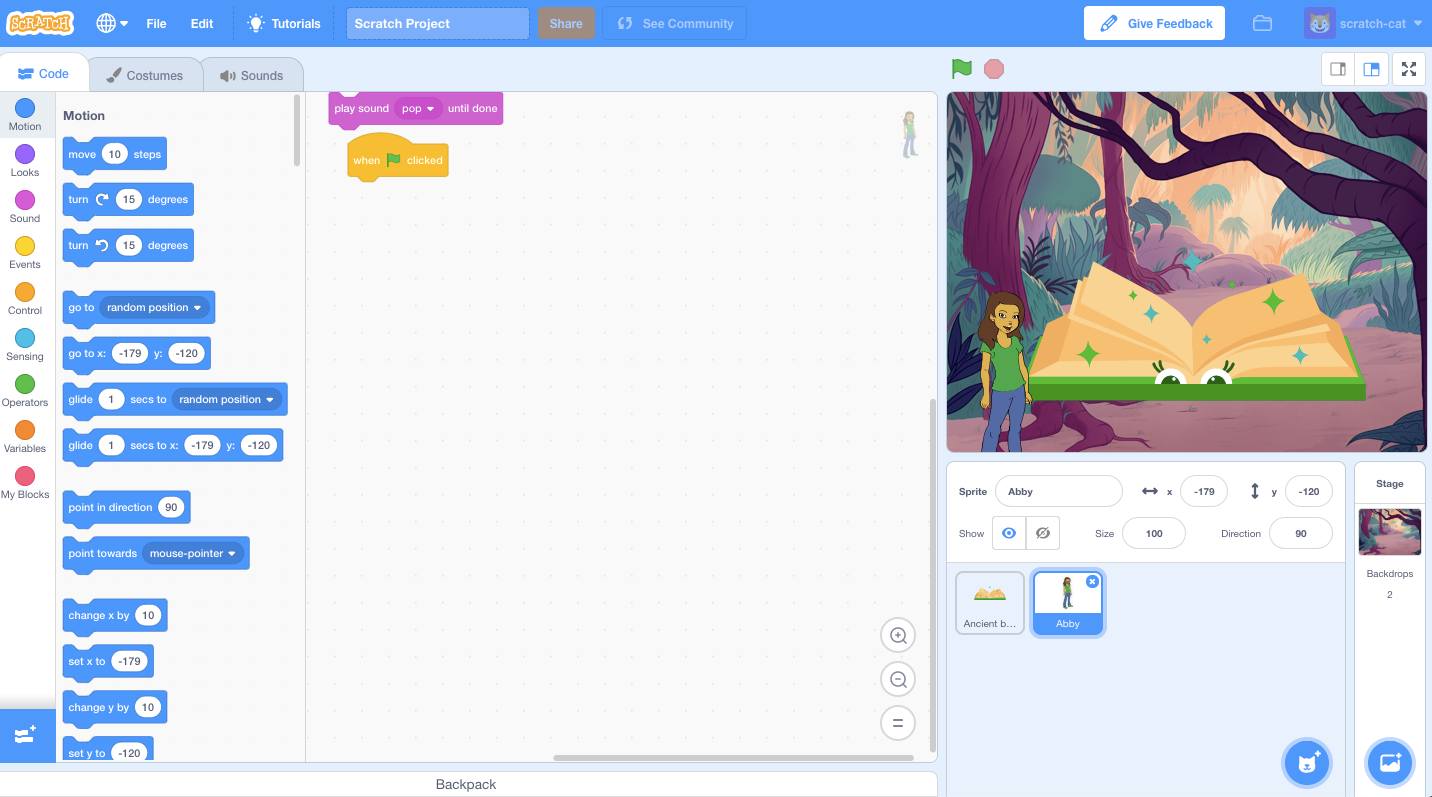
**Scratch Initial design blocks**



**Scratch project Interface**



**Visual Programming Language**



**Scratch Project Interface with Background**

**11. Evaluate the Designs:**

From the above Designs list, I am going to select one Interface Design for the next phase to enhance the project. In this I would like to select the Scratch programming Interface with Background because it is more attractive for students to select and play the game. Scratch has the significant highlights of many programming dialects, similar to circles (rehash squares) and conditionals (in the event that squares). With these highlights, Scratchers can make calculations, or directions to finish explicit undertakings. PC researchers would state Scratch is a Turing-complete programming language, which implies it can play out all the fundamental capacities that make up calculations. A few people stress that since you snap together squares in Scratch as opposed to composing content, is anything but a genuine programming language. Be that as it may, composing in content isn't a necessity for programming. Prior to the mid-80s, most developers composed and put away their projects utilizing openings on physical punch cards. Presently, many programming dialects are composed and put away with content, yet not all. For instance, engineers who utilize the programming language Labview make programs by interfacing graphical hubs. Scratch is one of many programming dialects on the planet. I conversed with Chris Garrity, a designer on the Scratch Team, whose present undertakings incorporate the Scratch site front end, ScratchJr, and confinement. In her profession, she has utilized many programming dialects; most as of late, she has for the most part worked in JavaScript, Ruby, Python, and, obviously, Scratch.

Creating a Rubric for Scratch Programming:

1. Get students to check out projects on the Scratch website and identify what qualities made for a good project. It is easy to find the good and bad projects because statistics are kept for each uploaded project. A popular project will have lots of love-its, re-mixes and downloads. See featured Scratch projects at the [project share website](http://scratch.mit.edu/channel/featured) .
2. Together with students list some of the characteristics of a good project. For example, some of the good project qualities: the project works well, easy to use, easy to understand, creative, fun to play, funny, cool, advanced scripts, cool sprites and backgrounds, creative drawings, and creative stories.
3. Use student feedback to create categories for evaluation. For example, project design/creativity, user friendliness, programming, backgrounds and sprites.
4. Come up with different levels of quality. For example: excellent, good, average and needs more work.
5. Create the rubric keeping in mind discussions of common problems and the qualities of good and not so good projects.
6. Using the freshly created rubric, chose several projects and evaluate them in groups or as a class.

**12. Conclusion:**

As mentioned earlier the purpose of this project is to develop a learning tool / Gaming App / Website to enhance the Knowledge of students. In this process I am going to develop a learning game for the concept of Monoalphabetic And Polyalphabetic Ciphers using the Scratch programming. Here I am going to select the Scratch project Interface with Background to grab the attention of students. By using this Design, I can say that it will definitely enhance the math skills of the students. I am going to include some games with the points concept and some animated designs to learn the concept in an ease way and to remember it in their brains forever. This conceptualized game will definitely improve their math skills. This Scratch programming language is a basic and easy for developing the gaming website mostly for the students. I chose this language because it is based on the blocked concept of code for the programming of the tool. I will be using this programming language and further developing or converting using a python or java or squeak programming code.