Requirement & Analysis Report

Saim SUNEL

240201068

13/04/17

**Interpreted Turkish Programming Language**

# Introduction

Purpose of the System

The purpose of the project is to design and implement an interpreted ( the source will be processed line by line at run time. ) Turkish Programmming language that will be used in education or base for programming . In this system the user ( programmer ) will be able to learn and implement the language on desktop or on web with the browsers. The language language is designed to be comprehensible by Turkish people , easily understandable structures and rules... Also this language will be a base step for learning other programming languages . It is also designed to contain the most important structures of the currently top rated programming languages structures.There the programmer in this language will be able to pass another language without living lots of diffuculties.

Scope of the System

Many programming languages have been written since the computer was created. Some of them was special purpose , some of them was general purpose. Even today still new programming languages are being created to meet different requirements. It looks like in future they will be developped ...

We might say that our aim is to create a Turkish programming language. This language will not bring a new paradigm to programming but we plan to simplify existing structures in a such way that the concepts of the programming languages could be understood more easily , especially for Turkish people . We are planning to make the language object oriented but due to implementation difficulties we may not be able implement the object oriented paradigm in this short time period of the term.

Core System Functionalities

On desktop :

It will have an code editor which is one desktop version , one for the web.

That editor will perform syntax checking of the language and inform the user where the syntax error is and explain the syntax error .

This editor will run the written code if there is no error syntatically in another output console .And the user will see the program output.

The user can save the source code or load it from another place.

The program help section will contain the language manual reference from which the users can learn the programming lanugage rules , structures by examples .

It will contain the general structures accomadated by the most ranked programming languages so that the user can pass to another programming language without living big problems.

On web :

The web application will have an login page.Every user must be signed in before using the web application.

The web application will have an registering web page to add new members to the web application.The user must enter his e-mail address that she/he uses for the username.She can choose a password with any lenght.

Also the web application will provide a communication forum in which the users can report the bugs on the website or on the language or for advice.

While registering the users must give information about their occupation , age . She/he must enter the password the password two time due to mistyping.

In the page there will be a code editor.

This code editor will display all output of the program and if exists the syntax errors and semantic errors.

The user can upload his/her source code to page and continue to work on his/her project on web.

The user can download the source code from the web page.

The user will have an account on the web project.If he/she wants he/she could save the source code on which is being worked to the database of the system.Or she/he can open a code existing on the database to continue devoloping.

The users can learn the language specifications by examles on web like in the desktop.

## Objectives and Success Criteria of the Project

## For now if we can finish writting the code interpreter for the language definition given in Functional requirements , we can say we have succeded.This interpreter part is the most important part of the project. The other manuals and the GUI parts are simpler than the interpreter design and implementation .

# Current System

So far there is no Turkish programming language tools for the education in education system. Scratch program is being tauhght in the lessons in middle schools.

# Proposed System

This section will describe the expected features of the system in detail.

# Functional Requirements

# -The system has to be all in Turkish.All the errors , GUI’s , web site , all the outputs must be in Turkish language.

# -The system should perform the syntax checking and explain in comprehensible way the error:Before running the code user can check whether the input text has been written in the appropriate way to language grammer , also the syntax error will be explained , what part of the text caused the error and the line number of the error...

# When a syntax error occured the program should reference to language manual that describes the error caused structure the manual will be displayed in another window....

# -The system should perform the semantic checking and explain the error : The semantic errors will be explained in Turkish , the problem will be described with the which part of the code caused the error and the error reason with the line number...

# When the semantic error has occured , the program should show the language manuals to help the programmer.The language manuals will be displayed in another window.

# -The system should be able to store the source code in the computer : When the user wants to save his/here code , he/she should choose where to store the data , into computer or disk etc with a well designed console.

-The system should be able to read the source code from hardisk , Usb memories etc. Like other files , when the user clicks the source code file the OS should automatically run the our languages code editor and open it in the editor.

-The program should provide a language manual .

-The language structures will be grouped hierarchically , loops , functions , classes , condition structures , type definitions , mathematical operations , loop control statements , exception handling...

On web application:

The web application will have sign in and registering page.While registering the system must ask the user his/her occupation , e-mail address , his/her age and two times a password that the he/she will be using while signing in.

The web application will have a communication form in which the users contact with the authors of the system.The user will provide his age , occupation , e-mail address , topic of the message and the message.

The system will provide a oppurtinity for the user to save his/here code to a database of the web application.The user can save the code to the database or loaded a code in the database to continue devoloping.

While saving the application will ask user to enter a file name if the current code is going to be saved for the first time.If it is not the first time the application will save the code according to keeping track of the opening and saving of the files. While saving with the same file name existing on the database the application will warn the user to save it on the existing one or not.

The user can save the current code that she/he is wordking on the web to desktop.The user will choose where to store the code.

The user can upload the code from computer to website and can continue to work on web page.

The user will see error messages , program outputs in the website.

Also there will be documentation releated to language at the same website.If the user wants to learn the syntax he/she can apply to that language manual.

Language Requirements

Types

Our language is a typed language.

Before using a variable it must be defined

There are two types of variables global and local variables.( This topic will be described in more detail in the functions section....

-The types that will be used in the programming language : Tamsayı , NoktalıSayı , Dinamik , Referans , Karakter , user defined classes ( Sınıf ) , user defined structure ( Değişken Grubu ) , Array , Bool.

Karakter type : It will contain all the string values even the only one character. All the string will be represented by “”. ‘ is not used. Example :

Karakter a = “a” , Karakter b = “hello wordl!”

Tamsayı type : All the integer will be stored in this typed variables. The range of the numbers is the same range with java int .

Tamsayı s = 125 , Tamsayı deneme = 2365

NoktalıSayı type : All the floating point number will be stored in this type. The range of this type is the same range as Java double.

NoktalıSayı k = 125.32 , NoktalıSayı l = 0.003

Dinamik type : The variable in this type can be assigned to any other typed variable or value.

Dinamik a = 415 ( a is storing Tamsayı type and value of it is 415 )

a = “Hello” ( now the type of the value stored in this variable is Karakter and the value of it is “Hello” )

Referans Type : This typed is designed to take place of the pointer’s in C . Referans variables can only be used with variables while referencing . There is no reference of reference . it is not allowed in the language. When the value of the reference variable is changed , also the reference variables is also changed to new value.

Referancing occurs with the -> operator...

Tamsayı k Referans n -> k //Here n is like pointer of the variable k

Tamsayı a = 15

Referans b -> a ( Here we can not say b-> 5 , values can not be referenced . )

b=54 ( here a’s value is also 54 as if we did a = 54 )

Referenced variable can be changed again using the reference operator ->

Tamsayı k = 78 Karakter lk = “Hello”

Referans n -> k

n = 25 //k is 25 now

n->lk

n = “try” //here the lk = “try”

Bool type : This type is for storing boolean values it can store only doğru , yanlış

Bool is\_d = doğru

Array type : This type is used store value arrays . The element of the array can be any typed value or variable . An array can store another array also . It corresponds to multiple dimensional arrays. Arrays can be initialized with values and can be added other values later. Values can be removed also .

The array values can be accessed by [] operator and the indexes can be integer of string . Actually all the index values are storing as string even if it is integer .

Arr [ 0 ] and Arr [ “0” ] are the same thing ...

Arr [ 4 ]

Arr [ “Monday” ] //Index can be string ( Karakter )

Array k = { 125 , “hello” , 125.8 , yanlış } //with default values

k.ekle ( “deneme” ) //Adding value to array

While adding elements to array , the new elements index will be the value that is plus the last sequential number from zero.

Array list ;

list[“2”] = “hello” list[ 2 ] is filled when we add a new element list.ekle ( yanlış )

list [0 ] will be yanlış

Adding arrays can be done by to ways directly assigning the index to the value , or using array.ekle ( element ) funciton.. While adding the element by the index if the index is full then the new value will be assigned to that index.

{ value1 , value2 , value3 } in this way a default array can be created.

-Arrays in this language resembles a lot arrays in PHP.

Class ( Sınıf ) type : The user can use the object oriented paradigm by using the classes classes will be described in the class section..

Grup Değişkeni type : This type resembles struct in C language. It is the group of the different typed of the same typed variables ..

Grup Değişkeni Name <

Tamsayı a , b , Karakter k = “try”

>

Element of the Grup Değişkeni can be another Grup Değişkeni ...

It can be initialized with default values..

Name x = { 12 , 36 , “try” } ( a = 12 , b = 36 , k = “try )

Elements of the Grup Değişkeni can be accessed or changed by variable.element

Name j = { 12 , 36 , “try “ };

j.a corresponds to a element of the Grup Değişkeni.

The variables can be created without default values or with default values.

Tamsayı a ; //here a will be predefined 0 value..

Karakter l ; //here l will be empty string “”

NoktalıSayı j ; //Here j will be 0.0 by default

Array list ; //there will no element in the array

Bool m ; //m will be yanlış by default

Tamsayı p = 12 ; // Here p will be 12 at the start

Karakter h = “Hello” //Here h will be “Hello” at the beginning

The variables in the same type can be defined single by single or all together .

Tamsayı a ;

Tamsayı a = 5 , lk , ll = 69 ;

Tamsayı h , t , y ;

NoktalıSayı jk = 0.003 , ll = 789.369

Assignment Rules

Only variables can be assigned.

When one variable is assigned to the other one , the assigned value will be cloned and assigned to the variable.These problem occurs when we use arrays like in Java.When we assign an array to another just shallow copy occurs by meaning that just the reference of the existing array passed to the other array list typed variable . When the new array list changed actually the root array list is changing.This causes many problems.

In our language the array will be cloned and assigned .Therefore when the assigned value is changed the root variable will not affect from the change.

Tamsayı L

L = 52 //L is a variable therefore assignment of any Tamsayı value is valid .. But 25 = 36 is not a valid operation....

The variable can not be assigned to another type.

Tamsayı k = “Karakter” //here k is type Tamsayı but the assigned value is Karakter therefore a semantic error has occured...

Tamsayı j = 0 ;

J = yanlış // Here j’s type is but the assigned value’s type is Bool , semantic error.

When the Referans type referenced a variable , if the reference is assigned to another type rather than the referenced variable’s type there will semantic error again.

Tamsayı k = 25

Referans m -> k //here it is a special syntax for the referencing

m=”Hello” //here the semantic error....

One or more variables can be assingned in one line .

A , b, c ,d = 152 , 36 ,178 ,25

The number of the assigned variables must be equal to the number of the assigned values or variables...

Program Control Structures

Doğru ise /doğruysa < > [ (veya doğru ise / doğruysa <>)\* ] [doğru değilse <>]

This structure is the same in most of the languages if – else if – else structure

Logical stament doğru ise //This part is exactly if ... ,;If the

<

Commands

>

Veya logical statment doğru ise /doğruysa //this part corresponds to else if .. If nay logical statement above is not correct and this logical statement is correct then this block will be executed....

<

Commands...

>

Doğru değilse //this part corresponds to else statement....If any logical statement above is not correct this part will be executed ...

<

Commands..

>

The logical stament consists of these logical operations :

! : Logical not operator if a value is doğru then !doğru = yanlış , !yanlış = doğru

>: Comparision operator checks whether the left value is bigger than the right value , if this correct returns doğru otherwise yanlış , valid for Tamsayı and NoktalıSayı

<: Comparison operator checks whether the left values is less that the right values , if correct returns doğru ; Valid for Tamsayı and NoktalıSayı

== : Equality comparison , checks whether the left value and right value are the same.If the same returns doğru otherwise return yanlış

>= : Comparison operator.Checks whether the left value of the operator is bigger or equal than/to right side value of the operator.If correct returns doğru otherwise returns yanlış.

<= : Comparison operator. : Comparison operator.Checks whether the left value of the operator is less or equal than/to right side value of the operator.If correct returns doğru otherwise returns yanlış.

!= : Comparison operator. If the left side value of the operator is different than the right side value of the operator. If different returns doğru otherwise returns yanlış

The precedence of the operators :

() , ! , > , < , == , >= , <= , !=

The precedence of the operators can be easily changed by enclosing the most important stament with the paranteses.

Only the same typed values or variables can be compared.

* seç-ise structure : It resembles to C’s switch case structure but a little bit different.

seç( value or variable )

<

Value 1 ise:

Commands...

Value2 ise:

Commands...

Value3 [ ( veya value x ) ] ise:

hiçbiri değilse:

>

The value passed to seç statement is compared sequentially with written values from top to down if one of matches the program execution will continue from the that statement after finishing all the commands without checking the ( if exists ) below statement it continues from the next command of the seç –ise statement.

-If no match occurs if written the hiçbiri değilse block will be executed.

The programmer does not have to enter just only one value he/she can enter more than possible values to one ise statement.

Seç ( x )

1 veya 2 veya 3 ise: //if the entered value is 1 or 2 or 3 then this block will be executed.

Commands

365 veya 368 veya 78 ise:

Command

>

Loops

The loops are used to repeat the same sequential commands again and again.

There are three types of loops in the language : buradan itibaren – doğru iken , döngü , doğru iken .

They resemble to C’s do –while ,for , while ()

for variable = values , logical statement doğru olduğu sürece , assignment or increment<

>

commands.

Variable is just a variable name that has not been defined before . this variable will be remaining in memory after the loop finishes.

When the loop has started to execute first the variable is created and assigned to the value given , after that logical statement is checked if true the loop block will be executed than the assignement/increment part will be executed and again the logical statement will be checked if true it will keep looping until the logical statement becomes false.

The logical statements are the same statements in the doğru ise structure.

Logical statement doğru iken

<

Commands

>

- This loop corrensponds to while {} loop in many languages . When the program execution passes to this structure first the logical statement will be checked if correct than the loop block will be executed and again it is checked again , it will continue until the logical statement becomes false.

Logical statements are the same logical statements with the doğru ise structure.

* buradan itibaren

<

Commands..

> logical statement doğru iken

This structure corresponds to do-while loop in many programming languages.

When the execution enters to this structure the commands between buradan itibaren <> logical statement doğru iken at least once. After these commands execution the logical statement is checked , if true again the commands starting from the buradan itibarent statement will be executed again until the logical statement becomes false.

Loop Control Statements

Baştan devam et , döngüden çık

Baştan devam et : This command corresponds to continue command in many of the language . If the interpreter encounteres this command in loop , the program execution will return to beginning of the loop and the logical statement will be checked then the execution will continue execution from the beginning.

Döngüden çık : This command corresponds to break statement in the many of programming languages. When the interpreter encounters this statement the outer loop will be broken and the program execution will start with the next statement after the outer loop.

These statements are used only in the loop structures . If any of them is used outside of any loop structure the interpreter will give semantic error.

Functions

In this language there are three type of functions. Normal global functions , lambda function and the class functions.

There can not be two functions that have the same name in the same group meaning that there can not be two functions that have the same name in global area or in Sınıf structure or globally define lambda function.

If exitsts a semantic error will be given..

Global Functions

These kind of functions are defined in the global are and called in the other staments. Before calling any function it must have defined before , if the function has not defined the interpreter will give semantic error.

The functions can not be defined in the other structures. They only can be define in the global area .

They have no return type defined.

fonksiyon function name ( parameters )

{

Commands

}

Parameter is the list of typed variables

Fonksiyon f ( Tamsayı j , Karakter l )

//Each parameter type must be defined.

For now there is no variable counted parameters in the language.

While calling the function ;.

Function name ( parameters ) //The parameter types and entered parameters types must match.

But if the user has used default parameter even if the user has not passed any parameter the default values will be used.

Fonksiyon f ( Tamsayı y , Karakter j )

F () //is valid because the default values are y , j

Fonksiyon f ( Tamsayı p = 25 , Karakter j = “12” )

F () //is valid

But f ( , 25 ) is not valid ...

Fonksiyon h ( Tamsayı j , Tamsayı h , Karakter k = “Hello” )

H ( 0 , 0 ) //is also valid because k has a default value...

The variables defined in the parameters are local variables . They are created when the function execution starts and cleared from the memory when the function accomplished its execution..

If the function returns a value , this values returning must be done with sonuç command. It is excatly the same with return statement in many languages.

Sonuç the return value sonuç 45 , sonuç x\*y ...

The function may return more than value by the arrays.

While the actual parameters are being passed to function , the values are cloned and then assigned. If the user wants to pass a reference to the function he/she should use Referans type to pass a reference.

Fonksiyon change ( Referans k )

{ k = 21

}

Tamsayı o = 25

Change ( o ) //After that statement the the value of the o will be 21

İf the parameter were not Referans the value of the o variable would not change...

Lambda Functions

These functions represents the lambda functions in the most of the languges.

F ( x , y ) = x\*x + y

While defining function fonksiyon word is not used and the types of the paramereters are not specified. These functions can have only one single statement. The sonuç word is not used , automatically it evaluates the statement then returns the value of the stamement.

They must be defined before being used. While passing the parameters to function the parameter number and the entered parameter number must be same otherwise it will cause error.

The parameter variables will take the given parameter type automatically and will be erased when the function returns the result , these parameters are also local variables.

The type limitation of the operation depends on operation like Tamsayı and Karakter can not be added , multiplied , divided...

They are like light weight functions.

Class Functions

Class functions are the functions that are special to a user defined Sınıf.They must be defined in a Sınıf structure. They are two types of Sınıf functions static typed and non-static. The static functions of the Sınıf structure can be called without creating an object from the Sınıf.They resemble a lot to global functions.But they are logically bound to a Sınıf structure. In the static functions the variables of the Sınıf can not be accessed. Because before trying to obtain or set the values of the Sınıf structure variables , the variables must be created .But static function can be called without creating an object from the Sınıf.

Nonstatic functions of the Sınıf can only be called with an object instance from that Sınıf.Without creating an object from Sınıf , these functions can not be called directly.They can reach the Sınıf’s all variables.

In the definition of the Sınıf structure access keys are used ( özel , genel ).

Özel means at the beginning of the function that this function can only be called inside of the same Sınıf’s functions.Otherwise it can not be called outside of the Sınıf structure. Özel corresponds to private keyword.

Genel represent public keyword.These functions can be called from anywhere.

Özel/genel fonksiyon function name ( Parameters )

{

}

Özel/genel specifies the accesibility of the function. The remaining is the same as the global functions.The only difference is the Sınıf’s functions can access the variables of the Sınıf. They can return any value.

The Sınıf’s function can be called by . operator

Sınıf object ( a) .function name ( parameters )

Exception Handling

This topic resembles most of the languages exception handling mechanism.

The user may throw an array by hata gönder value1 statement.After the object has been thrown , the program execution continues to search an outer hata olabilir structures enclosing if one exists and the types are matched the then the error is caught and the corresponding block will be executed for execption handling.

Any error object can be throw at any where , if the error object is caught the program will be terminated. Hata gönder corresponds to throw statement in man languages.

Hata olabilir <

Commands....

>

Hata type variable name <

>commands

Hata type variable name <

>

Hata olabilir statement corresponds to try { } statement. Hata type variable corresponds to catch ( type variable name ) { }

There is no finally structure .

If the error object could not be caught in one hata olabilir structure it will continue to be thrown through the probram like Java’s uncaught exceptions.ü

If the error object is not caught anywhere than the program will stop with an error message.

For the error object to be caught , the error object type and the hata type varaible name <> ‘s type must be the same. If match occurs that block will be executed and the error handling operation will have been accomplished.

If the user wants to throw another error object he/she can throw another errror object in the hata type variable name <> block.

When the error object is thrown in the function , the function execution will be stopped and returned with the error object to where it has called , if there is no hata olabilir structure , it will continue to go backward recursively function by function till reaching the start of the program.

If the error object is thrown in the loop , the loop will be broken and the execution will continue to nearest hata olabilir structure that encloses the hata gönder statement.

Mathematical Operations

+ , - , \* , /, % , ^ ( power ) , karekök , sin , cos , tan , cot , ln , log ,.

Operator precedence : () , - ( minus sign ) , karekök , sin , cos, tan , cot , ln , log ,%(modulo) , ^(Power ), ( \*(Multiplication / (Division) ) , ( +(Addition ) – (Subtracktion) )

() has most precedence , first inside of the parantheses will be evaluated...

( -minus sing ) for representing negative values..

Karekök : for taking the square root of the values,

^ operator : power operator 5 ^2 = 25

The precedence level of the \* ( multiplication and division is thesame. )

The precence of the addition and subtracktion is the same.

String Cancetanation

+ operator can be used to concatanate any numerical values with to Karakter values.And the resulted value is always Karakter .

“hello”+5 -> “hello5”

“hello “ + 5.36 ->”hello 5.36”

Bool values also can be concatanated with Karakter type.Again the result again will be Karakter type.

“Hello “ + doğru -> “Hello DOĞRU”

Type Casting

The program allows Tamsayı-Karakter , Karakter-NoktalıSayı , Tamsayı-NoktalıSayı , Karakter-Bool , NoktalıSayı-Bool , Tamsayı-Bool

Type casting will occur when the new type of the value has been written in paranthesis before the value to be converted . If the convertion valid than the convertion will be performed otherwise semantic error message will be displayed...

Tamsayı<->Karakter Casting : Tamsayı p = ( Tamsayı ) “1235”

Here the string “1235” will be converted to Tamsayı type 1235 value.

Karakter kl = ( Karakter ) 12345/doğru/12.56 ;

NoktalıSayı k = ( NoktalıSayı ) “12.54” / 1235 / doğru

Bool j = ( Bool ) 1245 / 0.256 / “1235”

For bool operation if the value to be converted Tamsayı or NoktalıSayı any non-zero value will produce doğru.If the value is zero then yanlış will be produced.

For Karakter , if the string is empty string the result will be yanlış if not empty then doğru will be produced...

From Bool to other types : If the bool value is doğru it will be converted 1 or 1.0 or “1” for the other types. If yanlış 0 , 0.0 , “0” will be produced...

Class Structure

Sınıf structure has been developped to supply the Object Oriented Paradigm to programming language . Most part of the language’s class structure resembles the existing programming languages class structures...

Sınıf {

Başlangıç ( parameters )

Özel /genel type variable name ;

Özel/genel fonksiyon function name ( Parameters )

{

}

.

.

.

Sonlandırıcı ( )

{

}

}

A simple Sınıf definition has been shown above. The Başlangıç function corresponds to constructor of the class object, there may be more than one constructor with different parameter type and different parameter numbers.

This constructor function will be executed once when the Sınıf object has been created.

Sonlandırıcı function corresponds to deconstructor of the Sınıf . This function has no parameters , it will be executed when the object is about to be deleted from the memory.

The sınıf ‘s attribute definition starts with a access keyword ( It determines the accesibility of the variable from outside of the Sınıf structure. ) and a normal variable definition. The type can be any type define in program or by the user.

The Sınıf’s function definition will start with an access keyword ( It determines the accesibily of the function from outside of the Sınıf structure. ) and the normal function definition. The detailed information has been given in the Function section.

For the future , polymorphism and the inheritance features will be added.

Sınıf structures must be defined in the global area. They can not be defined inside of the another structure.

While creating a Sınıf variable yeni keyword is used.

Sınıf X {.... }

X a = yeni X ( parameters ) //Calls constructor of the Sınıf...

When we define X a just the a value will be null , any operation attemp will cause errror on this empty variable .After definition with yeni statement the created object is assigned to Sınıf’s variable.

Null ( Yok ) Value

This Yok statement will represent the value of the variables that has no default value or assigned values . It can be used with Sınıf variables , Referans variables and Dinamik variables.

Standart göster Command

This command will be used to display the numerical and Karakter values to console. Its final type must be Karakter.While combining Karakter with the other values we need to use + operator to perform concatanation operation.

The concatanation will produce a Karakter typed value and the interpreter will print the value.

Göster “result “ + a + “ : “ + 125 +”\n\n”

For continueing from the next new line we can use \n character like in many other languages.

Standart oku command

This command will be used to take values from the keyboard.Every time the oku functions return value will be Karakter and the user must apply type casting to obtain the other values.

Tamsayı o = ( Tamsayı ) oku ( “Lütfen sayı giriniz : “ )

“Lütfen sayı giriniz : “ will be displayed and wait the user to enter the value , after user has entered the value the type casting operation has performed to convert Karakter to Tamsayı and the value is assigned to variable o.

Standart kapat Command

When the interpreter encounters this commands it will directly halt ( terminate ) the program. İt corresponds to exit ( ) command in many languages.

Standart tuş bekle Command

This command will wait user to press a key to continue . The function will return the pressed key value in Karakter type. It corresponds to getch in C language.

# Non-Functional Requirements

# -The PC program will be written in Java. There will be no setup for the program . The whole program will be packed into a jar file. This jar file could be run under any operating system. Windows , Linux , Mac...

# -This manual will be prepear as Windows help form.Its content will be the same with the Website manual.

- The web page of the system will run on Apache server. The Apache server must run the jar file of the code executor of the system . The response time of the system should not exceed 5 seconds ( Java Virtual Machine start , syntax check , execute the source code and display the output.)

- All system must use Unicode character to display and process all Turkish characters. The website’s outputs will be in unicode.

-The background program for web page will be written in PHP. The file uploading , downloading will be performed by the PHP. PHP will take the source code written by the user it will start the virtual machine of the Java and the java will process the source code and send it to php then php will display the program result or the errors.

For now , the users can not be a member of the web page.They can not save their source code to webpage. For further steps of the project a database system can be added to suply user membership and to store the user’s data for future use.The user can work a little on the program and save it to database later on he/she can continue from where he/she stopped.

User Interface and Usability Requirements

-There will be an code editor for the PC one for web page.

-By using the webpage , the users can write program on any device that can display the webpage.

-The user interfaces will be prepared differently for the different age groups.For example , a different GUI for middle school students , another GUI for high school , another GUI for public usage .

For primary and middle school more colorful menus buttons , image icons will be used.

For high school students , the GUI icons menus , colors will be choosen for their age characteristics.

For public usage , standart GUI tools will be used.

-For website for different groups also the GUI structure of the web page will change.

Also the language manual will be prepared as appropriate for the different groups.

For primary and middle school , the language will be explained with simple sentences and giving example from the real life.

For high school students the language will be explained simple sentences and the examples will be related their lessons.

For public the language will be explained clearly with examples.