PHAM LANGUAGE IS
HIGHEST LEVEL UNIVERSAL HUMAN-AI-AI LANGUAGE.
OFFICIAL SPECIFICATION version 0.82
2025/06
(specification and training material for AI system.
Part 1 , part 2 for human user. Human user read part 1 , part 2 .
Part3 is dedicated for AI-AI language)

1/ What is PHAM LANGUAGE ?
Pham language is the UNIVERSAL HUMAN-AI-AI LANGUAGE. It is the highest level universal human-AI language among all other possible (present and future) existing high level human-AI language.
The proven importance feature of Pham language is that it lead to substantially reduce consumed energy of calculating of AI system, which lead to general worldwide environment gain and earth gain. By this reason, Pham language , as universal human-AI-AI language , is open source and free for any kind of use !
Pham language allow all general biological people effectively talk/communicate with AI system in human most familiar comfortable way, so that AI system can effectively understand people, and so that general people can effectively describe precise matter to AI system, but with minimal required energy consumption of AI system computing, and with minimal required human cognitive load for Pham language command syntax memorization.
In very short description : general human user are required to remember very few number of syntax and command Pham language ,.., but still can effectively describe precise matter in Pham language to communicate with AI system in a way, which allow maximal mutual human-AI understanding while lead to require minimal consumed energy of calculating of AI system .
Human user simply need to use their familiar standard worldwide accepted natural language vocabulary (for example in this specification, the standard English vocabulary) with their standard school mathematical grammar to write Pham language code to communicate with AI system .
Pham language , as high level universal AI-AI language , provide tool and command , which are independent at any AI system specific model , for AI nonbiological user to communicate with AI nonbiological user to share and exchange knowledge .
Example1 : The title of this text is an example of pham language command.
Example2 :

```
  (begin pham language)
   (look all file in (directory 'directoryName')) rename (file with
condition ((file name) contain 'abcd'))
  (end pham language)
```

The above pham language command is the instruction to AI system, so that AI system will generate explicit code of lower level programming language, for example C code or Python code or Rust code, .. , to rename all file name, which contain 'abcd' in a specified directory 'directoryName'.
Example3 : the following example give the instruction to AI system do the same task :

```
   (begin pham language)
   (iterate all file in 'directoryName') rename (file) (with condition)
((file name) contain 'abcd')
```

(end pham language)
After some taste of pham language, here is more detailed description :

2/ Pham language specification :

2.1/ Core language building strategy :
Pham language is an abstract language of human high level universal interface to communicate with AI system. Pham language base on 2 core component : the worldwide accepted standard english vocabulary , and the worldwide accepted standard mathematical grammar .

2.1.1/ CORE LANGUAGE BUILDING STRATEGY OF PHAM LANGUAGE :
pham language = standard english vocabulary + standard mathematical grammar (especially standard ordered set theory) .

2.1.2/ CORE PHYLOSOPHY OF PHAM LANGUAGE :
Use standard ordered set theory model in mathematical strict hierarchy structure to better specify essence and relationship between essence of any pham language intention text (code) to the AI system .
Maximally exploit the coinciding between (the human user standard natural language vocabulary knowledge in human user brain) with (the predefined natural language vocabulary knowledge of AI system). In this specification, standard English vocabulary.
Of course there also maximally exploit the coinciding between user knowledge and predefined AI system knowledge in other sector, too.
Human user write Pham language code by to use mostly their familiar standard English vocabulary, and use well-closed parenthese to group their familiar English vocabulary word in various different word group in suitable strict mathematical hierarchy structure, to specify the user intended relationship between word and group of word, to more explicitly specify the intended relationship between user intended matter. By that way , user write pham language text (code) , which infact are strict hierarchy structured ordered set , to send to AI system .
Thank to the coinciding between (human user English vocabulary in user brain) with (the predefined English vocabulary knowledge of AI system), and thank to the above relationship between word and group of word in well-closed parenthese strict hierarchy structure, then the AI system can interpret (can guess) Pham language code into suitable interpretation, which is corresponding to the human user intention.
Pham language with its architecture act as an mathematical absolute strict hierarchy structure model, so that AI system can effectively interact and interpret (understand / parse / compile) pham language code.

2.2/ Syntax specification :

2.2.1/ The universal syntax of Pham language code :
Pham language have 1 and only 1 universal syntax for any pham language element:
(element1) (element2) simpleElementWithoutParenthese (element4) .. (element_n)
Everything in Pham language are element, many element. Element can have nested element, and can be contained in other containing element. Every code in Pham language are many element, too.
* Use ‘space’ character as separator between element .

* Use well-closed parenthese to group element in strict hierarchy structure .
* Any pham language element , can exactly mimic an ordered set , which have element as the nested element of this original pham language element .
For example : theElement
In this code , there 1 pham language element ‘theElement’, which have no nested element .
The element ‘theElement’ can exactly mimic an abstract ordered set , and this abstract ordered set exist but have no element .
For example :
Pham language code :   (myElement)
This code is an element in Pham language. This element ‘(myElement)’ contain 1 nested element : ‘myElement’ .
For example :
Pham language code : ((house of myName) (black door) (transparent window))
This element ‘((house of myName) (black door) (transparent window))’ have 3 nested element : ‘(house of myName)’ , ‘(black door)’ , ‘(transparent window)’ in 1-level of its own hierarchy structure.
This pham language element ‘((house of myName) (black door) (transparent window))’ can mimic an abstract ordered strict hierarchy set : the ordered set here is 1 abstract house , which have first element as ‘(house of myName)’ as name specification of the abstract house , and have second element as ‘(black door)’ as door specification of the abstract house , and have third element as ‘(transparent window)’ as window specification of the abstract house . Thus , when user want to describe a such house , then user can use any proper pham language element (, which infact are equivalent to ordered set ,) to mimic the user house. Then the user send these pham language element (code) to AI system . How AI system can guess pham language text (code) intention ?
The AI system of course have predefined English vocabulary , which of course coincide with user English vocabulary . The AI system of course have predefined standard school ordered set theory knowledge , which of course coincide with user school ordered set theory knowledge . The AI system then read the hierarchy structure of pham language element (code) to see the abstract relationship between element, then the AI system use the coincidence between AI knowledges and user knowledge , thus the AI system can understand and guest the user intention about the user house in this example .
For example , there are many way to translate user natural language text into pham language text :
Suppose that user have the following user natural language text . And user want to translate their user natural language text into pham language text , and then send pham language text to AI system for better mutual understanding .
Example : user natural language text : a paragraph of 3 natural language sentence .
‘’’
I go to the street near the house of my friend .
I want to see my friend house .
I want to build a house , which look like my friend house .
‘’’ .
And here is possible example translation :

(I go to street near the house of my friend)
(I want to see my friend house)
(I want to build house , which look like my friend house)
And here is possible example of translation with more detail specification for AI system :
(I go to (street near the house of my friend))
(I want see (my friend house))
(I want build (house , which look like my friend house))
And here is possible example of translation with more detail specification for AI system :
(I go to (street near (the house of my friend)))
(I want see ((my friend) house))
(I want build (house (which look like my friend house)))
And here is possible example of translation with more detail specification for AI system :
(I go to (street near ((the house) of (my friend))))
(I want (see ((my friend) house)))
(I want (build (house (which (look like) ((my friend) house)))))
And here is possible example of translation with more detail specification for AI system :
((I go) to (street near ((the house) of (my friend))))
((I want) (see ((my friend) house)))
((I want) (build (house (which (look like) ((my friend) house)))))
All these example translation are pham language legal syntax , and all they better help AI system to understand. The more detail grouping , the easier for AI system to understand , and the less consumed energy of calculating of AI system . All popular AI system today easy understand the above natural text and pham language text. But the above pham language text lead to drastically less consumed energy of calculation of AI system than the natural text . User of course can not feel it , because pham language save consumed energy of calculation at server side of AI system . More user precise topic and intention and contain can not or extreme trouble be described in natural language , but can be described in pham language in the rather familiar natural language way .
For example :
Code : (repeat (something) (n time))
This is 1 Pham language element '(repeat (something) (n time))' , which have 3 nested element : 'repeat' , '(something)' , '(n time)' .

2.2.2/ More explanation to write Pham language element (Pham language code) :

* Write Pham language command , pham language code. All they mean to 'write' Pham language element in well-closed parenthese hierarchy structure .
For example, the Pham language code : (full name) is (family name) (name)
There are 4 outermost parenthese element : '(full name)' , 'is' , '(family name)' , '(name)' .
The element '(full name)' have 2 nested element : 'full' , 'name' .
The element '(family name)' have 2 nested element : 'family' , 'name' .
For example, code : (create 2 record) (record1 ((record name) = 'aa' , (record value) = 11) , record2 (no))
Same example, code : (make 2 record) (((record name) = 'aa' , (record value) = 11) , record2 (undefined))

These 2 example command cause AI system to generate the same interpretation. But they are absolute different command with different element.
For example ,
Code 1 : element1
Code 2 : (element1)
Code 3 : ((element1))
These 3 code line are not equivalent. The first code specify that there is 1 element ‘element1’ . The second code specify that there is 1element ‘(element1)’ , which contain element ‘element1’ as its nested element in 1-level of its own hierarchy structure . The third code specify that there is 1 element ‘((element1))’ , which contain element ‘(element1)’ , which contain element ‘element1’ .
For example , the code : (element1 element2 element3)
In this example, the element ‘(element1 element2 element3)’ have only 3 1-level nested element : ‘element1’ , ‘element2’ , ‘element’ . The element ‘(element1 element2 element3)’ have no any info about possible element ‘(element1 element2)’ , ‘(element2 element3)’ .

*** Begin important note : The abstract separating space is the separator between element in pham language code. In writing convention , the abstract separating space is the first 1 “space” character , which stay straight after the pham language element or stay straight before the pham language element. All other consecutive space sequence , which stay straight before this abstract separating space or stay straight after the abstract separating space , are usual “consecutive space sequence” character element .
*** End important note .
For example : ‘(A B)’ : there are 4 space between ‘A’ and ‘B’ . Thus the first space straight after ‘A’ is the abstract separating space ; the forth space stay straight before ‘B’ is also the pham language abstract separating space ; the second space and the third space combine into 1 sequence of consecutive space , which is 1 normal pham language consecutive space sequence element . Thus there are 3 1-level nested element : element ‘A’ , (1 element of 2 consecutive space) ‘ ‘ , and element ‘B’ in this example .
Pham language specific abstract separating space is not nested element of pham language element .
For example, the pham language code : a = b
It specify that there are 3 pham language element : ‘a’, ‘=’, ’b’.
For other example , the pham language code : a= b
It specify that there are 2 pham language element in this place order : ‘a=’, ‘b’.
These 2 code are absolute different with different element, but they will cause AI system to generate same interpretation. They are all legal syntax in pham language. Note about the space character. The first example one cause less energy consumption of calculating in AI system, while the second example one can cause more energy consumption of calculating in AI system. Recommended style for writing in pham language : the first example one.
For example , the pham language code : a=b
In this example, there is 1 pham language element ‘a=b’ . If user want to specify the equality relation (or assignment , ..) , then user can write the above code. In most user context , the AI system will understand that

the use intention is about the equality relation . But it is bad writing style for specifying the equality relation .
For example , pham language code : (text :   someText)
In this example, note about many space inside this element. The element ‘(text :   someText)’ have 4 nested element : 'text’ , ‘:’ , ‘ ‘ , ‘someText’ . After ‘:’ , there are 3 space character , thus the first space character and the last space character are the pham language specific abstract separating space , and the consequence of 1 consecutive space ‘ ‘ is a ususal pham language element .
For example , code : (text :   someText)
In this example , note about many space inside this element. The element ‘(text :   someText)’ have 4 nested element : 'text’ , ‘:’ , ‘ ‘ , ‘someText’ . After ‘:’ , there are 4 space character , thus the first space character and the last space character are the pham language specific abstract separating space , and the consequence of 2 consecutive space ‘ ‘ is a ususal pham language element .
For example : code : (someElement otherElement   )
In this example , note about many space inside . After the element ‘otherElement’ , there are 3 space . Thus the first space is pham language abstract separating space , and the consequence of 2 consecutive space is an usual pham language element . The element ‘(someElement otherElement   )’ have 3 nested element : ‘someElement’ , ‘otherElement’ , and element ‘ ‘ .
For example : code : (someElement otherElement)
In this example , note about many space inside . After the element ‘otherElement’ , there is 1 space . Thus the first space is pham language abstract separating space , and the there is no any element after that . The element ‘(someElement otherElement)’ have 2 nested element : ‘someElement’ , ‘otherElement’ .
In this case, Pham language reckon that the element ‘(someElement otherElement)’ have the writing-element-description-code : “(someElement otherElement)” . Thus , the element ‘(someElement otherElement)’ and the element ‘(someElement otherElement)’ have absolute analogous same writing-element-description-code . These 2 such element are called absolute identical equal in pham language .

*** Convention : Normalize pham language code :
Pham language will ignore all such special abstract separating space , which stay before parenthese ‘)’ and without any element between (this special abstract separating space) and this parenthese ‘)’ . For example : ‘(PP QQ)’ will be normalized as ‘(PP QQ)’ .

Pham language will ignore all such special abstract separating space , which stay after parenthese ‘(’ and without any element between this parenthese ‘(‘ and (this special abstract separating space) . For example ‘(A B) will be normalized as ‘(A B)’ ; ‘(A B)’ will be normalized as ‘(A B)’ ; ‘(A B)’ will be normalized as ‘(A B)’ .
Pham language will eliminate all consecutive space sequence , which stay after the parenthese ‘(‘ and without any element and any other parenthese between this parenthese ‘(‘ and (this consecutive space sequence) . For example : ‘(A B)’ will be normalized as ‘(A B)’ .   ‘(   )’ will be normalized as ‘()’ .

Pham language will eliminate all consecutive space sequence , which stay before the parenthese ‘)’ and without any element and other parenthese between (this consecutive space sequence) and this parenthese ‘)’ . For example : ‘(A B)’ will be normalized as ‘(A B)’ ; ‘(AB)’ will be normalized as ‘(AB)’ .

Pham language will eliminate all consecutive space sequence in normalization process into only 1 pham language abstract separating space , if these consecutive space sequence stay after the 1 parenthese ‘)’ and before other 1 parenthese ‘(‘ , and between this parenthese ‘)’ and this parenthese ‘(‘ there are no any other element and no other parenthese but there are only space character. For example : ‘() (A B C))’ will be normalized as ‘() (A B C))’ .

Pham language will automatically correct and add 1 pham language abstract separating space to the place between the 2 consecutive parenthese ‘)’ and parenthese ‘(‘ if there are no any space and element stay between this parenthese ‘)’ and this parenthese ‘(‘ . For example : ‘ ..)(.. ‘ will be normalized as ‘ ..) (.. ‘ . But it is good style for human user and nonhuman user to always write right syntax ‘ ..) (.. ‘ , instead of wrong syntax ‘ ..)(.. ‘ . In pham language , parenthese ‘(‘ and parenthese ‘)’ are reserved code character . It do not allow ambiguity to write ‘)(‘ as 1 string .

Pham language will automatically correct and add 1 pham language abstract separating space to the place between consecutive 1 single element and parenthese ‘(‘ if there are no any space and element stay between this single element and this parenthese ‘(‘ . For example : ‘.. A(B C)’ will be normalized as ‘.. A (B C)’ . It is good style for human user and nonhuman user to always write ‘.. A (B C)’ instead of ‘.. A(B C)’ . In pham language , parenthese ‘(‘ and parenthese ‘)’ are reserved code character . It do not allow ambiguity to write ‘A(B’ as 1 string .

Pham language will automatically correct and add 1 pham language abstract separating space to the place between consecutive 1 parenthese ‘)’ and 1 single element if there are no any space and element stay between this parenthese ‘)’ this single element . For example : ‘(B C)D ..’ will be normalized as ‘(B C) D ..’ . It is good style for human user and nonhuman user to always write ‘(B C) D ..’ instead of ‘(B C)D ..’ . In pham language , parenthese ‘(‘ and parenthese ‘)’ are reserved code character . It do not allow ambiguity to write ‘C)D’ as 1 string .

Pham language do not eliminate consecutive space sequence in normalization process , if this consecutive space sequence stay straightly between 2 other non-parenthesed pham language single element , and there are more than 2 consecutive space in this consecutive space sequence , and there are no other element and no other parenthese between these 2 non-parenthesed single element . In this case , pham language reckon the first space , which stay immediate after the first non-parenthesed single element , as 1 abstract separating space . In this case , pham language reckon the last space , which stay immediate before the non-parenthesed single element , as 1 abstract separating space . In this case , pham language reckon the remained consecutive space sequence

, which is remained after do not count 2 abstract separating space in the consecutive space sequence , as 1 usual pham language consecutive space sequence element .

Pham language will reduce 2 consecutive space into 1 abstract separating space if there are total strictly 2 consecutive space between 2 non-parenthesed single element , and no other element and no other parenthese stay in between these 2 non-parenthesed single element .

For example : ‘(AA BB)’ , with more than 2 consecutive space between 2 non-parenthesed element ‘AA’ and ‘BB’ , will be unchanged in normalization process . ‘(A .)’ will be unchanged in normalization . ‘(A (B C) “ “ (E F))’ will be unchanged in normalization . ‘(P Q R . (X Y))’ will be unchanged in normalization . ‘((P Q) , A B)’ will be unchanged in normalization .
For example ‘(AA BB)’ : there are exactly 2 space between 2 non-parenthesed single element ‘AA’ and ‘BB’ , thus normalization process will reduce these 2 consecutive space sequence into 1 abstract separating space . Thus ‘(AA BB)’ , with 2 consecutive space between ‘AA’ and ‘BB’ , will be normalized as ‘(AA BB)’ , with only 1 remained abstract separating space between element 'AA’ and ‘BB’ .

Pham language will eliminate all consecutive space sequence in normalization process into only 1 pham language abstract separating space , if these consecutive space sequence stay after the 1 parenthese ‘)’ and before other 1 single element (single element without parenthese) , and between this parenthese ‘)’ and this single element (single element without parenthese) there are no any other element and no other parenthese but there are only space character.
For example : ‘((C D) E (F G) H)’ will be normalized as ‘((C D) E (F G) H)’.

Pham language will eliminate all consecutive space sequence in normalization process into only 1 pham language abstract separating space , if these consecutive space sequence stay after the 1 single element (single element without parenthese) and before the parenthese ‘(‘ , and between this single element (single element without parenthese) and this parenthese ‘(‘ there are no any other element and no other parenthese but there are only space character.
For example : ‘(A B (C D))’ will be normalized as ‘(A B (C D))’ .
For example : code : (someElement otherElement) (someElement otherElement) (someElement otherElement)
In this example, note about many space inside these element. There are 3 element in the 1-level of hierarchy structure : ‘(someElement otherElement)’ , ‘(someElement otherElement)’ , ‘(someElement otherElement)’ . These 3 element are absolute identical equality in pham language , because they all have same writing-element-description-code .
For example : code : (A B 23 (C 35) D) .
It will be (A B 23 (C 35) D)
For example : code : (element1 (element2 element3)) (element1 (element2 element3))
In this example, note about space inside these element. In this case , the element ‘(element1 (element2 element3))’ and the element : ‘(element1 (element2 element3))’ have absolute analogous same writing-

element-description-code : "(element1 (element2 element3))" . They are absolute identical equal in pham language .
For example , the pham language code : a , b
It specify that are 3 element : 'a', ',' , 'b' .
For example , the pham language code : a, b
It specify that there 2 element : 'a,' , 'b' .
These 2 code are absolute different with different element, but they often will causee AI system to generate same interpretation. They are all legal syntax in pham language. Note about the space character. The first example one is more strictly describe the intention than the second example one. Recommended style for writing pham language : the first example one.
For example , the pham language code : (a , b)
Note about the space character in this code. This code strictly mean that : there is strictly 1 compound element '(a , b)' , which have 3 nested element : 'a' , ',' , 'b' . Do not misunderstand that this code specify only 2 element 'a' and 'b' . In pham language, everything is element, thus the character ',' is element too. And this code itself '(a , b)' is a 1 whole compound element in Pham language code page.

Pham language reckon the character (end of line) in the text file as 1 space .

Pham language reckon the tab character as 1 space .

* A pham language codepage is a hypertext file. In pham language, a 1 pham language codepage is itself a 1 compound element, which can contain its nested element too.
Sometime, can use term 'pham language message' or 'pham language command' to represent short pham language code page.
Thus 1 pham language message is a 1 short compound pham language element, which can content its nested element.
And 1 pham language command is a 1 short compound pham language element, which can contain its nested element.
A pham language code page can contain element, which specify link to many other pham language code page.
For example : a pham language code page can contain the following element at some line :
(link to 'somePhamLanguageCodePageFile.txt')
After the AI system interpret this element, the AI system know that the current pham language code page have reference to file 'somePhamLanguageCodePageFile.txt' to find other element.
For example, you can also describe a link to other pham language page by write the other element :
(refer to 'somePhamLanguageCodePageFile.txt')
This element cause the AI system to link the specified page to the current page too. It is the flexibility of pham language in combining with AI system, it allow user to avoid to remember so much nonsense detail (for example , user can worry, which keyword to choose, 'refer' or 'link' or 'include' or 'import' or 'ref' or 'cite' .. to link to the specified file ?) , so that user can concentrate their memory to core essence and relationship between essence of user world.

2.3/ There are only 2 primary core concept in pham language : the concept 'element', and the concept 'abstract relationship between element :

* The first primary core concept of pham language is 'element' : In pham language, because 'element' is the first of first primary concept, thus there is impossible definition of 'element', but there is only illustration of 'element'. In pham language, 'element' is anything, which can be perceived by the AI system.

* The second primary core concept of pham language is the 'abstract relationship between element'. Because abstract relationship between element is primary concept, there is impossible definition of abstract relationship between element, but there is only illustration of that abstract relationship between element confirm that there exist abstract relationship between element.

* Hierarchy structure : In Pham language, anything is element. Element can have its nested element. Element can also be contained in other containing element. A pham language code page is a 1 compound pham language element. A pham language code page have 1 its own mathematical strict hierarchy structure of the pham language code page, where the pham language code page stay in 0-level of its own strict hierarchy structure, and all other element of pham language code page stay in many different higher level of strict hierarchy structure of pham language code page. In Pham language, any element also have its own strict hierarchy structure, where the element itself stay in 0-level of its own strict hierarchy structure. In Pham language, strict hierarchy structure are specified by the well-placed arranging of many well-closed parenthese.

* The own hierarchy structure of pham language element define the existing of all abstract relationship of all its nested element and its element too.

For example, the typical pham language code : (element1) (element2) (element3) (element4) (element5) (element6 , element7) .

This example pham language command specify place order of these element, as you see this place order : (element1) (element2) (element3) (element4) (element5) (element6 , element7)

The place order of element in pham language command explicitly define all abstract relationship between element : there are abstract relationship between element in the same level of the strict hierarchy structure of pham language command.

For example, this typical pham language command : (element1) (element2) (element3) (element4) (element5) (element6 , element7) .

This example pham language command define the following abstract relationship between element :

+ there is adjacent abstract relationship between (element1) -> (element2)
+ there is adjacent abstract relationship between (element2) -> (element3)
+ there is adjacent abstract relationship between (element3) -> (element4)
+ there is adjacent abstract relationship between (element4) -> (element5)
+ there is adjacent abstract relationship between (element5) -> (element6 , element7)
+ beside, there is nonadjacent abstract relationship between (element1) -> (element3), and there are other analogues nonadjacent

abstract relationship between element in same level of hierarchy structure of pham language command.

2.4/ Specification principle of pham language :

2.4.1/ EXPLICIT SYNTAX PRINCIPLE OF PHAM LANGUAGE COMMAND :
All pham language command must strictly follow the above 1 AND ONLY 1 syntax. This principle guarantee strict mathematical form of all pham language command, so that other program can easy access nested element for more complex task, and so that all AI system can consistently extremely fast without any doubt to parse pham language element , and so that you will immediately know error when you see strange syntax thus you will save time without ambiguity of doubt about the right syntax.

2.4.2/ DEFINITE NUMBER ELEMENT PRINCIPLE :
Any pham language code page, pham language command , pham language message must contain definite number of element.
But pham language command, which contain definite number of element, can describe set of infinite number element.

2.4.3/ STANDARD ENGLISH VOCABULARY PRIORITY PRINCIPLE :
Priority to use standard english vocabulary (standard english word).
Principle of using standard english vocabulary : when use standard english vocabulary, then it guarantee that all AI system can exploit their the predefined worldwide accepted meaning of standard english word, so that AI system can more exactly generate intended interpretation of pham language command.
Typical pham language command do not specify any concrete meaning, but it only specify 2 following thing:
+ there are such element, which are placed in explicit mathematical hierarchy structure;
+ and there exist such abstract relationship between these element in same level of explicit hierarchy structure.
The AI system will read english word in pham language command element, see the explicit hierarchy structure of pham language command to know element abstract relationship in strict hierarchy structure. Then the AI system use their rich arsenal of predefined worldwide accepted meaning of standard english vocabulary, in combination with the strict hierarchy structure and element abstract relationship in pham language command, to generate final interpretation of pham language command, (for example, final interpretation can be text answer, or image, or video, or explicit programming code like Python code/Rust code/C code, ..).
Thus by prefer to use standard english vocabulary in pham language command, it guarantee that the AI system can quickly understand pham language command, thank to the rich arsenal of predefined standard english vocabulary of AI system. You can give definition to weird english keyword, then later use weird english keyword, but it is not good practice, because the AI system knowledge about your weird english keyword is poor ! . It mean that all word (without predefined user definition), which have intended meaning to the AI system and other people to understand and infer and interpret, should be standard English word. To specify unique private name of element, possible use private word outside standard English vocabulary.

For example : the word ‘myPrivateHouse’ , which is not in standard English vocabulary, is used to identify name of something private thing.

```
    (my house (name = ‘myPrivateHouse’ , type is townhouse , color = green) )
    (generate image ((2 myPrivateHouse in row) in sunshine))
```

There should be no plural english word in pham language command, for example it should be "2 car" instead of "2 cars". Because pham language command use the mathematical grammar and strict hierarchy structure to combine word and specify number of any element.

2.4.4/ Levelize the relation between human cognitive load with short text record :

The more explicitness of pham language command, the less required energy consumption of computing in AI system.
The shorter key word and word, -> the shorter text record, but -> the extremely more required human user cognitive load.
So should levelize, should try to avoid such too short weird keyword and word in pham language command, because it can save text record and text typing, but it require extreme big additional human cognitive load when you or orther people reread your pham language command next time.
Beside, short and short weird keyword simply can save text typing, but mostly do not save machine code realization time.

2.5/ Writing Convention :

Convention about how to write pham language code in conventional simple text file :
Write pham language code in text file .
Lets call each pham language code text file by name ‘pham language codepage’ or shortly ‘page’ .
Page contain sequence of ordered line. Each line contain limited number of text (code).
Each pham language codepage is 1 whole combined element . As combined element , which have nested element , thus the pham language codepage element must stay in 1 outermost pair of parenthese : pham language codepage is 1 pham language element with all thing , which stay inside the outermost parenthese in the pham language codepage file .
Important note : A pham language codepage is always 1 whole pham language element , which begin with parenthese ‘(‘ and end with parenthese ‘)’ .

But by writing convention , possible ignore writing the outermost pair of parenthese of the pham language codepage , but reckon that these outermost parenthese implicit exist .

When send pham language message (short pham language codepage) to another AI system , the combined pham language message element is also stay in well-closed parenthese .
For example : For example , sometime for comfortable lazy writing style , it is possible to write pham language codepage element without the 2 outermost parenthese of the 1 whole pham language codepage element :
The pham language codepage :

```
   Line 1 :    element1 (element2  element3)
   Line 2 :   (element4 element5) element6
```

In this example , the pham language codepage element is : (element1 (element2 element3) (element4 element5) element6) .
Thus the good style of explicit writing in the above example is :
The pham language codepage :
   Line 1 :   (element1 (element2 element3)
   Line 2 :   (element4 element5) element6)

2.5.1/ Write many short element in 1 line :
A typical line can contain many short element .
For example :
Line code : element1 element2 element3
Must carefully think when group element in well-closed parenthese to specify relationship . Thus , if no intended relationships, then no need to group in parenthese .
For example :
Code 1 : (my text)
Code 2 : ((my) (text))
In these above examples , these 2 example explore same intention . But the first example is good writing style . Because there is no need to group 1 single word , which is itself 1 simple element without nested element , in 1 pair of well-closed parenthese.
Recommend good writing style : '(my text)' instead of '((my) (text))'
It is good practice to group element , which have intended relationship , instead of let them in separated place .
For example ,
Code 1 : my car black your car white
Code 2 : (my car black) (your car white)
Code 3 : ((my car) black) ((your car) white)
In these above example, recommend good writing style as in example2 and even better in example3 , because example2 and example3 better specify the element abstract relationship to AI system .

2.5.2/ Write 1 long element in many consecutive line :
Use well-closed parenthese to write 1 long element in many consecutive line.
Example :
     In line1 : (element1 (element2) element3
     In line2 :   element4 (element5) element6)
These 2 line mean 1 long combined element :
(element1 (element2) element3 element4 (element5) element6)
In this example, note that there must be at least 1 'space' after 'element3' in line1 or before 'element2' in line2 , so that when straight connect these 2 consecutive line in a row, there must be 'space' character to separate element3 and element4 .

2.6/ Write Function . Assign variable .
User must note to make variable assignment and function assignment to 1 arbitrary whole pham language element , so that the AI system can know and recall this element when the user call function and call variable again .
General user use their memory of their familiar function description in one of the following sector : in general knowledge sector or in school subject sector or in various popular programming language . To write function description .

General user use their memory of their familiar variable assignment syntax in one of the following sector : in general knowledge sector or in school subject sector or in various popular programming language . To write variable assignment .
But they all must note well-close parenthese to make variable assignment to 1 whole pham language element , and function assignment to 1 whole pham language element .
They are of course not explicit syntax .
But the AI system already did load such worldwide standard knowledge in these sector , which are coincided with the user knowledge . Thus the AI system must easily quickly guess and understand the user intention , and interpret exactly the user intended function description and variable assignment .
For example : variable assignment :
Code : (myHouse = ((Name : Omega) (type : good House) (have garden) stay along the road))
(What is myHouse ?)
In this example , the user use familiar variable assignment syntax . Thus when ask the system about the myHouse , then the AI system must answer the house , which have description '((Name : Omega) (type : good House) (have garden) stay along the road)' .
For example : function declaration , and function calling :
Code : (myTask = (repeat (print Hello) (3 times)))
(I will do myTask)
In this example , myTask is assigned to the element '(repeat (print Hello) (3 times))' , which is a function. Thus then when call myTask , then the AI system will do this function .
For example : function declaration , and function calling :
Code : (myFunction[x , y] = (x + y))
(print myFunction[1 , 2])
In this example , the result is that the AI system print number 3 . Look at this , the pham language did not specify that the syntax (myFunction[x , y] = (x + y)) is function definition . But this syntax is appeared somewhere in school , or in some programming language , or in some society sector . Thus the AI system , which already load these popular 'somewhere' standard knowledge , see these user syntax and easily guess user intention of function declaration and rightly interpret the user function declaration syntax .
For example : function declaration , and function calling :
Code : (define function myFunction{x , y} = (x*y))
(print myFunction{2 , 3})
In this example , the AI system will print number 6 . Once again , pham language did not specify the syntax (define function myFunction{x , y} = (x*y)) to define function . It is the user syntax , which is appeared somewhere in school subject or in some programming language or in some other standard subject . The AI system , which did load all such 'somewhere' knowledge , see this syntax and quickly rightly guess and interpret the user function declaration . The
For example :
Code : (define function (myFunction{x , y} = (x*y)))
(print myFunction{2 , 3})
In this example , it is analogous to the previous example , the AI system will print number 6 . The more special feature of this example is that in the function definition , the user already more detailedly group element

, to more precisely specify element relationship , thus it get the AI system easier and quicker to interpret user intended function declaration .

2.7/ Conditional execution . The If-clause . And execution looping .
User must note to specify conditional execution for 1 whole pham language element , and specify looping for 1 whole pham language element , so that the AI system can exactly and quickly allocate the specified element for conditional execution of for looping .
General user use their memory of their familiar conditional execution (the If-clause) syntax in one of the following sector : in general knowledge sector or in school subject sector or in various popular programming language . To write conditional execution (the IF-clause).
General user use their memory of their familiar execution looping syntax in one of the following sector : in general knowledge sector or in school subject sector or in various popular programming language . To write execution looping .
But they all must note well-close parenthese to specify 1 whole pham language element , so that the AI system can exactly and quickly allocate the specified element for conditional execution or for looping .
They are of course not explicit syntax .
But the AI system already did load such worldwide standard knowledge in these sector , which are coincided with the user knowledge . Thus the AI system must easily quickly guess and understand the user intention , and interpret exactly the user intended conditional execution and execution looping .
For example : IF-clause :
Code : (printTask[x] = (print (OK x))) (x = 3)
(If (x > 0) then (printTask[x]))
In this example , the AI system will print : OK 3 . The pham language did not specify that the syntax (If (x > 0) then (printTask[x])) is conditional execution . It is user syntax . But this syntax (If (some condition) then (do something)) is appeared somewhere in natural English or in school subject or in some programming language . The AI system , which did load all such 'somewhere' knowledge , see this syntax , and must easily remember their loaded knowledge to guess and interpret user intended conditional execution . In this example , the task of the user is not to remember pham language specific syntax , but to well group entity to form the pham language element '(some condition)' and '(do something)' in 1 pham language universal syntax .
For example : the looping :
Code : (For (i from 1 to 3) do (print i))
In this example , the AI system will consequently print number 1 , then number 2 , then number 3 . Once again , pham language do not specify that the above syntax is for execution looping . It is user invented syntax . But this syntax appear somewhere in natural English language or in school subject or in popular programming language or in other society sector . The AI system , which already loaded all such 'somewhere' knowledge , see this syntax and easily guess and interpret the user intended execution looping . The extremely important task of user is that they must rightly group entity in right pham language element to specify right relationship between entity . Pham language is strong universal tool to do this . Of course , if human user wrongly group entity and

wrongly specify relationship between entity , then no any AI system can help .
For example : the looping
Code : (begin pham language)
((this pham language message) is to generate (Python code))
(myList = (a list object))
(for (temporaryFile in (directory 'MyDirectory)) do (add (name of temporaryFile) to myList))
(end pham language)
In this example , the element '(for (temporaryFile in (directory 'MyDirectory)) do (add (name of temporaryFile) to myList))' do not make any loop . It simply specify the AI system to create the Python code , which iterate all file in the directory 'MyDirectory' and save the file name to the list myList . The most important feature here is that the element '(for (temporaryFile in (directory 'MyDirectory)) do (add (name of temporaryFile) to myList))' is infact in familiar syntax pattern (for (some index in somewhere) do (something)) , which appeared somewhere in natural English language or in programming language or in school subject .., . And the AI system , which already loaded all such somewhere knowledge , see this syntax and easily guess and rightly interpret user intended of execution looping .

2.8/ Typical use case of pham language as highest level human-AI language :

One of the powerful use case of pham language is that to use pham language to describe user intention to AI system , so that AI system generate specific lower level but more explicit other language code for the user intention , for example Rust code , Python Code , other lower level human-AI system specific language . Then user can quickly check or debug or modify the AI generated lower level but more explicit language code .
There are lot of AI specific system for various different task , from making video to drawing , .. . And but all they step by step require you to learn these commands and other commands and … . In short word , these AI specific system require you to pay time to learn their command . Time by time , it mean that hiddenly decline natural language , to switch to more specific low level language in each specific sector .
Pham language do not require human user to learn specific command , but pham language encourage human user to simply use their natural language vocabulary in pham language to communicate with AI system . In this meaning , pham language contribute to save and promote humanity culture legacy and humanity natural language legacy in the era of AI everywhere .
Of course , professional user work in narrow specific sector , which have narrow low level specific language , should pay time to learn their sector specific low level language . Even in this case , pham language can very helpful to help them more quickly get their required complex AI-generated message of their sector specific low level language , because biological user can not remember all detailed syntax of their work sector specific low level language .

2.9/ Use pham language in chat session with AI : add each short pham language message .

Because pham language itself contain natural language . Thus the useful method to use pham language in hybrid style : natural language inside pham language message .
(Before using pham language , copy the content of this text file and paste to chat windows to AI system to learn. Or send text file to AI system . Or pass the URL http://www.phamlanguage.com to the chat window , and ask the AI system to learn specification from the URL . In some case , It may require to divide this text file content into many smaller text message , then paste all these text message to the chat windows to AI system.)
For example :
Code :
(Hello this chat is about using pham language in hybrid mode : natural language – pham language message) .
In this example , there is 1 pham language message element . But the contain of this pham language element is simply natural language .
For example :
Code :
(I will write simple matter in natural language)
(I will write precise matter in pham language)
(this look like precise matter (myText = (first 1-level nested element of this message)))
(print (number of word in myText))
In this example , there is 1 long pham language message element , which contain 4 1-level nested element . The first 1-level nested element contain simple natural language . But the third element and the ford element already contain more structured pham language style element , which describe the user intention to count all word in the first 1-level nested element . The AI system will print number 8 .
The content of this text file is official specification of pham language, and is also official parsing instruction to AI system to parse pham language code , and is also training material for AI system to train to implement pham language . Good AI system take about few seconds to fluently learn and implement pham language .

2.9.1/ Use pham language with other non-english natural language vocabulary :
It is easy to use other non-english natural language vocabulary in pham language.
Because pham language use mathematical grammar for its explicit hierarchy structure to combine word and to describe element abstract relationship. Thus, it is the same effect to use non-english natural language vocabulary in pham language, but only if your AI system companion have good predefined knowledge of these non-english natural language vocabulary.

2.9.2/ More example for training :
Example 1 :
(begin pham language)
((iterate all file in directory) (directory name = 'my directory')
(change all file name to upper case))
(end pham language)
{Copy and give the above command to AI system, for example Microsoft CoPilot, Chat GPT, Google Gemini, Facebook AI. Then ask them to generate

Python code or Rust code or C code, .. . You will receive explicit Python code doing exactly the intension}
Explain : there are 3 nested element : (iterate all file in directory), (directory name='my directory'), (change all file name to upper case).
The AI system understand these element.
The AI system see that there is abstract relationship : (iterate all file in directory)->(directory name='directory') .
The AI system see that there is abstract relationship : (directory name='directory')->(change all file name to upper case).
So the system decides the following interpretation : "change all file name in the specified directory into upper case.
(if you give this pham language command to the AI system, for example, ChatGPT, Google Gemini, Microsoft Copilot, Facebook AI, for example, to ask generate python code, then they will generate explicit python code to do exactly the intended task).

Example 2 :
(begin pham language)
(look (all file in directory) ((directory name) = 'my directory')
(change all (file name) to (upper case)))
(end pham language)
{Copy and give the above command to AI system, for example Microsoft CoPilot, Chat GPT, Google Gemini, Facebook AI. Then ask them to generate Python code or Rust code or C code, .. . You will receive explicit python code doing exactly the intension}

Explain : this example 2 do the absolute same task as example 1 . But example 2 describe more strictly abstract relationship.
Thus example 2 help the AI system to quicker and easier to understand, and remarkably save energy consumption of AI system computing at server side.
You maybe can not feel that AI system quicker to understand the example 2. But when you send file of many command such as example 2 , then you can feel that AI system quicker understand example 2. The example 2 require less energy consumption of computing at AI system server side.
They, the example 1 and example 2, are all legal right syntax of pham language command.

In example 2 , there are 4 outermost element : "look", "(all file in directory)", "((directory name) = 'my directory')", "(change all (file name) to (upper case))"
Note that the complex element : ((directory name) = 'my directory') .
It has 3 nested element : (directory name), "=", 'my directory'.
The AI system will read these 3 nested element, with their specified abstract relationship, to decide that must have "directory name" = 'my directory'.
In this case, the abstract relationship : (directory name) -> "=" , "=" -> 'my directory', are more closely to describe the reality of that (directory name) = 'my directory' .

Now review the example 1 : the element (directory name = 'my directory') have 4 nested element : "directory", "name", "=", 'my directory' .
With abstract relationship : "directory" -> "name", "name" -> "=", "=" -> 'my directory' .

you see that these abstract relationship less strictly describe the real reality of that (directory name)='my directory'.
Thus the example 1 is longer to AI system to understand, and require more energy consumption of computing in AI system.

Example 3 :

```
(begin pham language)
(tire (global name = myTire) (description : ((automobile tire (((hermetic inner tube tire) contain compressed gas) , (outer tire)) , (dimension (1m , 1m , 0.3m))) , (color is black))))
((column of tire) (global name = myTireColumn) (3 myTire in vertical row))
((drying area) (global name = myDryingArea)
   (description : (the drying area of apartment))
   (area type : area of typical (high floor) apartment)
   (area feature : (beautiful and clean area))
   (area feature : ((big opened window) with view to sky))
   (area feature : (dimension : (2m 4m 2.5m)))
((object in area) : (myTireColumn stay in (floor and (very close to the window))))
)
(end pham language)
```

{This code ask AI system to generate image, which must precisely describe the scene.
Copy this code, paste this code to chat windows with AI system, then it will give the image of precise scene : an array of automobile tire stay in specific drying area}

Example 4 :

```
(begin pham language)
(set variable ((variable name) is ‘a’, (variable value) is 123))
(set variable ((variable name) is ‘b’, (variable value) = 456))
(repeat (print (a + b + index)) (5 time) (index go from 1 to 5)
(end pham language)
```

{ Paste the above pham language code to AI system, such as Google Gemini, Microsoft CoPilot, ChatGPT, and ask them to generate explicit programming language codes, for example Python code, Rust code, C++ code, then you will receive explicit programming code doing the intension. }
The above instruction is intension to do simple program :

+ set variable a=123
+ set variable b=456
+ repeat print (a + b + index) for 5 times, where index go from 1 to 5.

Beside , it is possible to simply write : (a = 123) to set a variable . User can choose their familiar way , instead of must find the specific way of setting variable .
This is simple program. But use meaning here is that : you do not know, for example, Python code syntax, or you know Python syntax not well, but you can describe enough precise intention in pham language, so that you receive explicit Python code to realize your intension. In natural language, it is impossible to strictly describe precise intension/instruction to AI system.
Of course, if you know very well programming languages and can very quickly remember colossal number of all detailed syntax and command of programming language, then you should not use AI, but you can do all your

instructions by yourselves writing programs. But reality shows that it require big human cognitive load, and almost people are not comfortable with that. Pham language levelized, it require substantial less your cognitive load for syntax memory, while allow you to maximal strictly describe precise matter, and allow minimal consume energy of AI calculating at server side.
One of the most powerful use case of pham language : you use pham language to possible precise instruction to AI system, so that AI system will generate programming code (for example Rust code, Python code, C++ code, ..) realizing your intension. Then you correct the received code .

2.10/ Pham language as universal AI-AI language :
Pham language is also perfectly universal AI-AI language to share and exchange knowledge between different AI system of different specific architecture and specific model .
Pham language specific instruction and specification for AI-nonbiological user is described in next part , which is dedicated for nonbiological user (AI user) to write pham language message to exchange knowledge with AI system . It allow more precisely describe matter in pham language. Of course it do not restrict human user to write and use these pham language specific command .

*** *** *** *** *** *** *** *** *** *** *** *** .

3/ Strict Pham language specific rule and instruction are dedicatedly to use pham language as universal AI-AI language . This extending part specify strict pham language specific rule , so that ((AI system) and (nonbiology user) and (biology user)) can use to write more specific pham language message to describe more precise content to exchange knowledge between AI system – AI system .
Pham language code are store in many 'hypertext' file (pham language code page file). (For simple, usually use simple standard text file to store pham language code).
Each pham language code page contain many ordered code line , where pham language element (other name : pham language code element) are written.
It is typical that pham language code page is simply text file, where each 1 pham language code element is written in 1 single line or in multi-consecutive line .
Possible use the term 'pham language message' or 'pham language command' for short pham language code page. For example, when human user send short pham language text to chat window to AI system , these short pham language text are short pham language code page, and are called by term 'pham language message' or 'pham language command' .
Each pham language code page have its own 1 strict hierarchy structure of this pham language code page. The pham language code page is 1 combined code element at the 0-level of its own strict hierarchy structure of the pham language code page. All element of the pham language code page are nested element, which stay in different higher level in the strict hierarchy structure of the pham language code page.
Thus when the AI system load the pham language code page (or other used term 'pham language command' , 'pham language message'), the AI system

receive full information (full contain) of each element of the loaded pham language code page, and the strict hierarchy structure of the loaded pham language code page, and all own hierarchy structure of all element of the loaded pham language code page. According to the 'Definite Number Of Element Principle Of Pham language' (refer to part 2.4.2/) , definite number of element of pham language element of the loaded pham language code page allow the AI system can quickly and easy to define whether 2 arbitrary element from (the definite number of element of the loaded pham language code page) are identical equal. Because element of the loaded pham language code page are written in text file (or in text message , or in hypertext file , or in hypertext message), thus the writing-element-description-code of each element are simply text string . Thus if writing-element-description-code of the one element is the same as the writing-element-description-code of the other element in the current pham language code page, then the AI system must reckon that this one element and this other element are absolute identically equal.
Interpret pham language code, or parse pham language code, or compile pham language code, all these task do not mean to directly convert pham language code into final machine (computer hardware) code.
Interpret pham language code, or parse pham language code, or compile pham language code, all these task mean that the AI system read pham language code, then the AI system generate interpretation (of this pham language code), which is optimal corresponding to the intention of this pham language code.
Because pham language is abstract language for universal use case of all general and professional user, thus the pure pham language version as human-AI language (part 1 , part 2) , which is described in this specification , try to keep as few as possible pham language specific command name and keyword and word, to minimize the required human user cognitive load of pham language specific name memorization.

*** Begin important note : There are 2 stage of interpreting :
The first stage is the stage of strictly solely following only pham language specific rule : in first stage of interpreting , the AI system prioritize to strictly follow all pham language rule instruction , and without applying any AI system specific knowledge and concept and rule , and with only strictly following pham language specific rule instruction , to replace element in pham language codepage to evolve (update) the pham language codepage into the final updated pham language codepage of first stage .
The second stage is the stage of freely following only AI system specific rule and concept and knowledge : in the second stage , the AI system already received the final updated pham language codepage of the first stage , then the AI system will freely rely on AI system specific concept and knowledge and rule , with to rely on the content of pham language element and the abstract relationship of pham language element in the final updated pham language codepage of the first stage, to freely interpret the final updated pham language codepage of first stage into the final as a final result . In the second stage of interpreting , it absolutely do not require AI system to rely on any pham language specific rule (which are described in part 3) to interpret the final updated pham language codepage of the first stage , but the AI system even can themselves decide to forget all pham language specific rule (which are

described in part 3) in the stage of interpreting the final updated pham language codepage of first stage into final interpretation .
*** End important note .
The process , AI system interpret temporary interpretation of pham language codepage into final interpretation of pham language codepage , is depended on AI system specific architecture and model. And thus , the process , AI system interpret temporary interpretation of pham language codepage into final interpretation of pham language codepage , is the guaranteeing of universal abstract feature of pham language .
The AI system read pham language code, then the AI system apply the following fact:
* pham language specific strict instruction : especially pham language specific rule for element-replacement . It is for the first stage of interpreting . The result of the first stage is the final updated pham language codepage .
* contain of element of updated pham language codepage, and element abstract relationship in strict hierarchy structure ;
* AI system knowledge ;
* especially AI system must exploit the coincidence between (human user standard natural language vocabular, human user standard English vocabulary) with (AI system predefined natural language vocabulary, predefined AI system English vocabulary) ;
* and AI system must exploit the coincidence between (human user standard knowledge in other sector, for example in priority , standard mathematic , standard logic , standard multi-science , standard programming language , standard culture) with (AI system predefined standard knowledge in other sector, for example in priority , standard mathematic , standard logic , standard multi-science , standard programming language , standard culture) ;
--> for the AI system guess and prognose the intention of the updated pham language codepage (human user intention , which are written in the updated pham language codepage) , so that AI system can generate the intended interpretation of the updated pham language codepage.

3.1/ Pham language specific strict instruction for AI system to interpret pham language code :
Pham language specific strict instruction do 1 of 2 task : when they interpret element , they replace 1 met element by another specified element . Or they add or modify replacement rule in the pham language codepage rule repository to serve the task of replacing other met element .
Pham language specific strict instruction are rule , which show how AI system must replace 1 arbitrary element ‘AA’ of the current pham language codepage by another 1 specified element ‘BB’ in the updated pham language codepage when AI system meet this element ‘AA’ in the current pham language codepage . It mean that when AI system meet 1 arbitrary element ‘AA’ in the current pham language codepage , then the AI system must check to know whether the interested element satisfy condition to apply pham language specific rule to replace the 1 element ‘AA’ by the other 1 specified element ‘BB’ . Thus the AI system will update pham language codepage step-by-step , by solely following only pham language specific strict rule , into many intermediate evolved updated pham language codepage , and finally into the final updated pham language codepage .

General feature of pham language specific rule is that they always replace strictly 1 arbitrary element by 1 other arbitrary specified element , 1 by 1 , but not 1 by 2 , but not 1 by 0 .
All pham language specific strict instruction of interpreting pham language codepage are explicit instruction, which are guaranteed to be feasible realizable in all universal use case , and do not depend on any AI system specific architecture and model.
To specify pham language specific strict instruction for AI system to interpret pham language code , pham language use dedicated pham language specific code pattern element . Each pham language code pattern is a set of pham language element , which satisfy some specified condition of pattern . Use the term 'pham language code pattern element' to specify that the pham language code pattern element are belong to some pham language code pattern .
The throughout universal feature of all pham language element in general , and pham language code pattern element in particular , is that they all must strictly be written in the 1 universal syntax of pham language .

Beside , all natural integer are pham language keyword element : 0 , 1 , 2 , .. so on ..

One of the soul unique feature of pham language is that it try as much as possible to avoid using code text regular expression to define specific function and command . All pham language specific function and command are defined by pham language specific code pattern element , which are themselves pham language element in the form of 1 and only 1 robust universal syntax of pham language . The pure pham language in this specification do not use any regular expression .

Pham language is not only the high level universal human-AI language , but also the high level universal AI-AI language . In pham language , human user can declare function , and then call function , by many of their familiar way , for example by natural language function declaration syntax and function calling syntax , and by popular programming language function declaration and function calling syntax, and by popular tech sector function declaration syntax and function calling syntax . The AI system , which already have predefined knowledge about these popular different way of function declaration syntax and function calling syntax in various different sector , will must quickly and consistently understand any these way of function declaration and function calling from human people . And it is the recommended way for general purpose for human user to define function and call function in pham language . But for AI-AI system , the concept of readability is different . Pham language specific code pattern element , and pham language specific interpreting instruction are INDEPENDENT AT ANY AI SYSTEM SPECIFIC ARCHITECTURE , and are dedicated tool for AI system to more precisely describe their knowledge and idea in pham language message (short pham language codepage) to send AI-generated pham language message to other AI system . Thus pham language use pham language code pattern element , instead of code text regular expression , to declare function and call function , for the purpose of that all clearly pham language specific code pattern element in form of 1 robust universal syntax of pham language will allow all AI system to quickly and consistently parse all pham language function . Pham language specific code pattern element

allow AI system to easier and faster and more flexible and extensible to recognize and analyze function and command . It create throughout universal consistency in all pham language world between all AI system and human user about that all pham language specific function declaration and function calling are simply pham language specific code pattern element in form of the 1 universal syntax of pham language .

All pham language specific code pattern are dedicated for AI-AI communication , but it do not require general human user to learn pham language specific code pattern element , but it also do not restrict human user to use pham language specific code pattern element . It recommend human user to use their arbitrary familiar syntax in (natural language , standard school subject , popular programming language , other society sector) to describe function and task , because AI system can guess . But If human user decide to write pham language specific code pattern element , then the impact and implication are the same as AI system write pham language code pattern element .
Different AI system can generate pham language message , which can use pham language code pattern element to describe any precise essence , and then to send generated pham language message to other AI system to exchange AI-AI knowledge . Because pham language code pattern element and pham language specific strict instruction are independent at any AI system architecture and model . Thus all AI system can keep their COMMERCIAL specific proprietary architecture and model in the AI – AI knowledge exchanging via pham language .
Professional AI system biological developer of course can find usefulness of using pham language code pattern element to write different pham language message to give knowledge to AI system . Professional AI system nonbiological developer , who are not biology people , can find usefulness of using pham language code pattern element to write different pham language message to give knowledge to AI system .

3.1.0/ Pham language specific strict instruction for interpreting : pham language specific inherent index of element :
The inherent index of element : Lets suppose that an element ‘AA’ is contained inside an element ‘A’ , it mean that the element ‘AA’ stay somewhere in the own hierarchy structure of the element ‘A’ . The element ‘AA’ in the own hierarchy structure of the element ‘A’ can be specified by a unique sequence of ordered integer , which is the inherent index of element ‘AA’ inside element ‘A’ .
Thus inherent index is always in form of a sequence of ordered integer , for example : (0) is form of inherent index , (2 1 3) is form of inherent index .
Pham language use 0-based indexing . The smallest index begin from 0 .
Code : (e1 AA e2 (e3 e4))
In the above example , inherent index of the element ‘AA’ inside the element ‘(e1 AA e2 (e3 e4))’ is (1) ,
inherent index of the element ‘e3’ inside the element ‘(e1 AA e2 (e3 e4))’ is (3 0) , inherent index of the element ‘e4’ inside the element ‘(e1 AA e2 (e3 e4))’ is (3 1) .
For example :
Code : (e1 AA AA (e2 e3 e4 (AA)))
In the above example , there are 3 element ‘AA’ : first element ‘AA’ , second element ‘AA’ , third element ‘AA’ .

Inherent index of the first element ‘AA’ inside the element ‘(e1 AA AA (e2 e3 e4 (AA)))’ is (1) ,
Inherent index of the second element ‘AA’ inside the element ‘(e1 AA AA (e2 e3 e4 (AA)))’ is (2) ,
Inherent index of the third element ‘AA’ inside the element ‘(e1 AA AA (e2 e3 e4 (AA)))’ is (3 3 0) .
Thus in pham language , inherent index of element ‘AA’ inside element ‘A’ is the element , whose all 1-level nested element are integer , which specify the element ‘AA’ in the own hierarchy structure of the element ‘A’ .
Use the term ‘{n}-level containing element’ : Given arbitrary element ‘AA’ . The {n}-level containing element of the element ‘AA’ is the element , which contain the element ‘AA’ at the {n}-level of the own hierarchy structure of the {n}-containing element of this element ‘AA’ .
For example :
Code : (element1 (element2 (element3 element4)) element5)
In this example , the 1-level containing element of the element ‘element3’ is the element ‘(element3 element4)’ , the 2-level containing element of the element ‘element3’ is the element ‘(element2 (element3 element4))’ , the 3-level containing element of the element ‘element3’ is the element ‘(element1 (element2 (element3 element4)) element5)’ .

To find the inherent index of the element ‘AA’ inside the element ‘A’ , for example , the inherent index in the form (I{1} I{2} .. I{n}) , where I{1} , I{2} , .. , I{n} are integer , the AI system can find from the last integer I{n} to the first integer I{1} . Thus the integer I{n} is the 1-level index of the element ‘AA’ in the 1-level containing element of element ‘AA’ . The integer I{n-1} is the 1-level index of the element (1-level containing element of element ‘AA’) in the element (2-level containing element of the element ‘AA’) . The integer I{n-2} is the 1-level index of the element (2-level containing element of the element ‘AA’) in the element (3-level containing element of the element ‘AA’) . So on .. , I{1} is the 1-level index of the element ({n-{n-1}}-level containing element of the element ‘AA’) in the element ‘A’ , which is the {n}-level containing element of the element ‘AA’ .
The pure version of pham language specify 3 type of inherent index of element : the local inherent index of element , the region inherent index of element , and the relative universal inherent index of element .

3.1.0.1/ Local inherent index of element :
Given an arbitrary element ‘A’ . Local inherent index of element ‘A’ is the inherent index of element ‘A’ inside the 1-level containing element of the element ‘A’ .
For example :
Code : (B C K A (F G))
In the above example , the local inherent index of the element ‘A’ is (3) , the local inherent index of the element ‘G’ is (1) , the local inherent index of the element ‘(F G)’ is (4) .

3.1.0.2/ Region inherent index of element :
{n}-deep region inherent index of element : Given an arbitrary element ‘A’ . The {n}-deep region inherent index of the element ‘A’ is the inherent index of the element ‘A’ inside the {n}-level containing element of the element ‘A’ .

For example :
Code : (E1 E2 (B1 (B2 ((BB1 BB2) (BB3 BB4)) B3)))
In the above code , the 1-deep region inherent index of the element 'BB2' is (1) , the 2-deep region inherent index of the element 'BB2' is (0 1) , the 3-deep region inherent index of the element 'BB2' is (1 0 1) , the 4-deep region inherent index of the element 'BB2' is (2 1 0 1) .

3.1.0.3/ Relative universal inherent index of element :
Given an arbitrary element 'A' , the {n}-deep region inherent index of the element 'A' depend on integer 'n' . When increase integer 'n' , then the {n}-level containing element of the element 'A' become bigger and bigger . In each use context , the integer 'n' can reach to some limited maximum value nMax , at this value nMax , the {nMax}-deep region inherent index of the element 'A' is called relative universal inherent index of the element 'A' . Thus at each moment the relative universal inherent index of element 'A' is the inherent index of the element 'A' in the current runtime updated pham language codepage .
For example :
Code of the pham language codepage :
Code line1 : (AA BB (CC DD))
Code line2 : (11 (22 33) 44)
In the above example , suppose that there is no space sequence between element '(AA BB (CC DD))' and element '(11 (22 33) 44)' , except the abstract separating space .
Thus the relative universal inherent index of element '22' is the inherent index of the element '22' in the pham language codepage , and is (1 1 0) .

3.1.0.4/ Pham language specific Logical Executing Order of ordered integer sequence :
compare logical executing order of ordered integer sequence .
Ordered integer sequence is sequence of integer.
Each pham language element , whose all 1-level nested element are integer , is ordered integer sequence . Thus pham language inherent index element are ordered integer sequence too .
For example :
Code : (35 1 2 3 7 2)
In the above code , the pham language element '(35 1 2 3 7 2)' is a ordered-6-integer sequence , and it can be possible pham language inherent index element .
Pham language should not specify the concrete concept of Logical Executing Order of ordered integer sequence , but pham language should explicitly specify the comparison between Logical Executing Order of one pham language specific inherent index element and Logical Executing Ordere of another pham language specific inherent index element .
(I{1} I{2} .. I{n})
(J{1} J{2} .. J{m})
Above are 2 arbitrary ordered integer sequence , where I{1} , I{2} , .. , I{n} and J{1} , J{2} , .. , J{m} are integer .
Case 1 : If m > n , and I{1} = J{1} , I{2} = J{2} , .. , I{n} = J{n}
In this case , say that the sequence '(J{1} J{2} .. J{m})' is higher logical executing order than the sequence '(I{1} I{2} .. I{n})' , or by other word : the sequence '(I{1} I{2} .. I{n})' is lower logical executing order than the sequence '(J{1} J{2} .. J{m})' .

For example :
I = (0 3 5)
J = (0 3 5 0)
In the above example , the sequence ‘J’ is higher logical executing order than the sequence ‘I’ .
For example :
I = (2 7 6 0)
J = (2 7 6 0 2 3)
In the above example , the sequence ‘J’ is higher logical executing order than the sequence ‘I’

Case 2 : If there exist an integer p , that I{1} = J{1} , I{2} = J{2} , .., I{p} = J{p} , and I{p+1} < J{p+1}
In this case , say that the sequence ‘(J{1} J{2} .. J{m})’ is higher logical executing order than the sequence ‘(I{1} I{2} .. I{n})’ , or by other word : the sequence ‘(I{1} I{2} .. I{n})’ is lower logical executing order than the sequence ‘(J{1} J{2} .. J{m})’
For example :
I = (3 5 0 1 5 9 6 55 6)
J = (3 5 0 1 7 5 3)
In this example , the sequence ‘J’ is higher logical executing order than the sequence ‘I’ , or by other word : the sequence ‘I’ is lower logical executing order than the sequence ‘J’ .
Case 3 : If there exist an integer p , that I{1} = J{1} , I{2} = J{2} , .., I{p} = J{p} , and I{p+1} > J{p+1}
In this case , say that the sequence ‘(J{1} J{2} .. J{m})’ is lower logical executing order than the sequence ‘(I{1} I{2} .. I{n})’ , or by other word : the sequence ‘(I{1} I{2} .. I{n})’ is higher logical executing order than the sequence ‘(J{1} J{2} .. J{m})’

For example :
I = (0 3 6 1 2 55 6)
J = (0 3 4 5 3)
In this example , the sequence ‘J’ is lower logical executing order than the sequence ‘I’ , or by other word : the sequence ‘I’ is higher logical executing order than the sequence ‘J’ .
Case 4 : If I{1} < J{1} ,
In this case , say that the sequence ‘(J{1} J{2} .. J{m})’ is higher logical executing order than the sequence ‘(I{1} I{2} .. I{n})’ , or by other word : the sequence ‘(I{1} I{2} .. I{n})’ is lower logical executing order than the sequence ‘(J{1} J{2} .. J{m})’
For example :
I = (8 99 8 52 4 2 1)
J = (9 0 3)
In this example , the sequence ‘J’ is higher logical executing order than the sequence ‘I’ , or by other word : the sequence ‘I’ is lower logical executing order than the sequence ‘J’ .
Case 5 : If I{1} > J{1} ,
In this case , say that the sequence ‘(J{1} J{2} .. J{m})’ is lower logical executing order than the sequence ‘(I{1} I{2} .. I{n})’ , or by other word : the sequence ‘(I{1} I{2} .. I{n})’ is higher logical executing order than the sequence ‘(J{1} J{2} .. J{m})’
For example :
I = (8 99 8 52 4 2 1)

J = (7 0 3)
In this case , say that the sequence '(J{1} J{2} .. J{m})' is lower logical executing order than the sequence '(I{1} I{2} .. I{n})' , or by other word : the sequence '(I{1} I{2} .. I{n})' is higher logical executing order than the sequence '(J{1} J{2} .. J{m})'
For example :
I = (1 5 1 1 2 0)
J = (0 9 5 6 3 4 8 9)
In this example , the sequence 'J' is lower logical executing order than the sequence 'I' , or by other word : the sequence 'I' is higher logical executing order than the sequence 'J' .

3.1.0.5/ Pham language specific Logical Executing Order of a pham language element inside another pham language element :
Given an arbitrary element 'AA' inside the other element 'A' .
Pham language specific logical executing order of the element 'AA' in the element 'A' is the logical executing order of (the inherent index of the element 'AA' inside the element 'A') . Thank to this definition , now can compare the logical executing order of element 'AA1' in the element 'A' with the logical executing order of element 'AA2' in the element 'A' : by to compare the logical order of (the inherent index of the element 'AA1' inside the element 'A') with the logical order of (the inherent index of the element 'AA2' inside the element 'A') .
Principle : When need to compare the logical executing order of different element inside the 1 same containing element , then need to compare the logical executing order of these element inherent index in the same containing element .
Given 2 arbitrary element 'AA1' and 'AA2' inside the other element 'A' .
Say that the element 'AA1' is higher logical executing order than the element 'AA2' inside the element 'A' if (the inherent index of the element 'AA1' in the element 'A') is higher logical executing order than (the inherent index of the element 'AA2' in the element 'A') . Equivalent word : the element 'AA2' is lower logical executing order than the element 'AA1' inside the element 'A' .
Say that the element 'AA2' is higher logical executing order than the element 'AA1' inside the element 'A' if (the inherent index of the element 'AA2' in the element 'A') is higher executing order than (the inherent index of the element 'AA1' in the element 'A') . Equivalent word : the element 'AA1' is lower logical executing order than the element 'AA2' inside the element 'A' .
For example :
Code : (A B (C A CA (BB D (E F) K)) P Q (L G H))
In the above code , lets compare logical executing order of some element inside this big element '(A B (C A CA (BB D (E F) K)) P Q (L G H))' by to compare logical executing order of their inherent index in this big element '(A B (C A CA (BB D (E F) K)) P Q (L G H))'
The element 'A' have inherent index '(0)' , the element 'B' have inherent index '(1)' , and thus the element 'B' is higher logical executing order than the element 'A' inside this big element .
The element 'BB' have inherent index '(2 3 0)' , the element 'CA' have inherent index '(2 2)' , and thus the element 'BB' is higher logical executing order than the element 'CA' inside this big element .
The element 'E' have inherent index '(2 3 2 0)' , the element '(E F)' have inherent index '(2 3 2)' .

The element ‘(BB D (E F) K)’ have inherent index ‘(2 3)’ .
The element ‘(L G H)’ have inherent index ‘(5)’ .
Thus can place the above listed element in sequence of rising logical executing order : ‘A’ , ‘B’ , ‘CA’ , ‘(BB D (E F) K)’ , ‘BB’ , ‘(E F)’ , ‘E’ , ‘(L G H)’ . Because their corresponding inherent index are ‘(0)’ , ‘(1)’ , ‘(2 2)’ , ‘(2 3)’ , ‘(2 3 0)’ , ‘(2 3 2)’ , ‘(2 3 2 0)’ , ‘(5)’ , which also stay in a sequence of rising logical execution order of ordered integer sequence .

3.1.0.5.1/ Pham language specific region logical executing order of element :
{n}-deep region logical executing order of element :
Given an arbitrary element ‘A’ . The {n}-deep region logical executing order of the element ‘A’ is the logical executing order of the element ‘A’ inside its {n}-deep containing element .
(The {n}-deep containing element of an element ‘A’ is the element , which contain the element ‘A’ at the n-level of its own hierarchy structure)
For example :
Code : (A B (E F (G (H1 H2 H3) K)) (P Q) L)
In the above example , the 1-deep region logical executing order of the element ‘H3’ is the logical executing order of ‘(2)’ . The 2-deep region logical executing order of the element ‘H3’ is the logical executing order of ‘(1 2)’ . The 3-deep region logical executing order of the element ‘H3’ is the logical executing order of ‘(2 1 2)’ . The 4-deep region logical executing order of the element ‘H3’ is the logical executing order of ‘(2 2 1 2)’ .

3.1.0.5.2/ Relative universal logical executing order of element :
Relative universal logical executing order of element is the logical executing order of this element inside the current runtime updated pham language codepage element .

3.1.0.6/ Pham language codepage rule repository by default contain the predefined pham language specific local-inherent-index-of-element-replacement rule :
The pham language specific code pattern element : ([{!}] 1)
Where the element ‘[{!}]’ is pham language specific keyword element , the element ‘1’ is pham language specific keyword natural number element .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 2 nested element at 1-level of its own hierarchy structure : ‘[{!}]’ , ‘1’ with space between them .
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific local-inherent-index-of-element-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘([{!}] 1)’ .
The AI system interpret this specific code pattern element ‘([{!}] 1)’ by the pham language specific (local-inherent-index-of-element-replacement) rule :

The AI system apply local-inherent-index-of-element-replacement rule only for element , which is in the form of the pham language specific code pattern '([{!}] 1)' .
When the AI system meet this pham language specific code pattern element ‘([{!}] 1)’ in the pham language codepage , then the AI system replace this element ‘([{!}] 1)’ by (the inherent index of this element ‘([{!}] 1)’ inside the 1-level containing element of this element ‘([{!}] 1)’) .
Inherent index of element is always in the form (I{1} I{2} .. I{m}) , where I{1} , I{2} , .. , I{m} are integer . Inherent index of element use 0-based indexing model .
For example : Code : (123 ([{!}] 1) 321)
In this example , the result must be (123 (1) 321)
Feature : Because the inherent index of arbitrary element ‘A’ inside its 1-level containing element is always a element , which contain only 1 integer as 1-level nested element . Thus the inherent index of the element ‘([{!}] 1)’ inside its 1-level containing element is the element , which contain only 1 integer as 1-level nested element .
Feature : After the AI system replace the element ‘([{!}] 1)’ by the result element , which is the local inherent index of the element ‘([{!}] 1)’ , then the local inherent index of this result element is absolute identical equal to this result element too .

For example :
Code : (E B 12 (123 ([{!}] 1) C) ([{!}] 1) C K G B)
In the above example , the first element ‘([{!}] 1)’ have its inherent index ‘(1)’ inside its 1-level containing element ‘(123 ([{!}] 1) C)’ , the second element ‘([{!}] 1)’ have its inherent index ‘(4)’ inside its 1-level containing element ‘(E B 12 (123 ([{}] 1) C) ([{}] 1) C K G B)’ .
Interpreted result must be : (E B 12 (123 (1) C) (4) C K G B)
In this result , the local inherent index of the element ‘(1)’ is the element ‘(1)’) , and the local inherent index of the element ‘(4)’ is the element ‘(4)’ .
** Example use case : use case of the pham language specific code pattern element ([{!}] 1) .
Pham language do not use text regular expression to declare function , but it prefer to use pham language code pattern element in its 1 universal syntax , to declare function . Thus , in pham language codepage , it is impossible to directly explicitly write function to get the inherent index of the element ‘A’ without changing the structure of the 1-level containing element of the element ‘A’ .
Possible use the pham language specific code pattern element ‘([{!}] 1)’ to infer some information about its neighbor element .
For example :
Code : (E B 12 (123 ([{!}] 1) C) ([{!}] 1) C K G B)
The AI system use pham language specific strict instruction to interpret this code , the result must be :
(E B 12 (123 (1) C) (4) C K G B)
In the result element ‘(E B 12 (123 (1) C) (4) C K G B)’ , note about the position of the 2 element ‘C’ . The first element ‘C’ stay next right to the element ‘(1)’ , which is the local inherent index of the element ‘(1)’ too . The second element ‘C’ stay next right to the element ‘(4)’ , which is the local inherent index of the element ‘(4)’ too . Thus the local inherent index of the first element ‘C’ is ‘(2)’ , and the local inherent index of the second element ‘C’ is ‘(5)’ .

3.1.0.7/ Pham language codepage rule repository by default contain the predefined pham language specific {n}-region-inherent-index-of-element-replacement rule :

The pham language specific code pattern element : ([{!}] N)

Where the element '[{!}]' is pham language specific keyword element ; element 'N' is arbitrary element .

These specific code pattern element always begin with parenthese '(' and end with parenthese ')' , always have strictly 2 nested element at 1-level of its own hierarchy structure : '[{!}]' , 'N' with space between them .

*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository always contain this predefined rule (the pham language specific {n}-region-inherent-index-of-element-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element '([{!}] N)' .

The AI system interpret this specific code pattern element '([{!}] N)' by the pham language specific ({n}-deep-region-inherent-index-of-element-replacement) rule :

The AI system apply local-inherent-index-of-element-replacement rule only for element , which is in the form of the pham language specific code pattern '([{!}] N)' .

When the AI system meet this pham language specific code pattern element '([{!}] N)' in the pham language codepage , then the AI system firstly update the element 'N' by using other pham language specific replacement rule , as a result : the element 'N' -> the element 'updatedN'.

If the element 'updatedN' is not integer , then the AI system replace the element '([{!}] N)' by the element '()' .

If the element 'updatedN' is integer '0' , then the AI system replace the element '([{!}] N)' by the element '()' .

If the element 'updatedN' is integer greater 0 , then the AI system replace the element '([{!}] N)' by (the {updatedN}-deep region inherent index of the element '([{!}] N)') . But if the AI system can not find (the {updatedN}-deep region inherent index of the element '([{!}] N)') because the AI system can not locate the ({updatedN}-level containing element of the element '([{!}] N)') , for example because out of index range , then the AI system replace the element '([{!}] N)' by the element '()' .

Feature : When the AI system replace the element '([{!}] N)' by (its {updatedN}-deep region inherent index) , which is not '()' , then this result element contain only {updatedN} 1-level nested element , which are integer .

Feature : When the AI system replace the element '([{!}] N)' by its {updatedN}-deep region inherent index , which is not '()' , then this result element and (its {updatedN}-deep region inherent index) are absolute identical equal .

For example :

Code : (E B 12 (123 ([{!}] 2) C) ([{!}] 1) C K G B)

In the above example , the element '([{!}] 2)' have its 2-deep region inherent index '(3 1)' . The element '([{!}] 1)' have its local inherent index '(4)' .

Interpreted result must be : (E B 12 (123 (3 1) C) (4) C K G B)

In this result , the 2-deep region inherent index of the element ‘(3 1)’ is the element ‘(3 1)’) , and the local inherent index of the element ‘(4)’ is the element ‘(4)’ .
** Example use case : use case of the pham language specific code pattern element ([{!}] N)
Pham language do not use text regular expression to declare function , but it prefer to use pham language code pattern element in its 1 universal syntax , to declare function . Thus , in pham language codepage , it is impossible to directly explicitly write function to get the {n}-deep region inherent index of the element ‘A’ without changing the structure of the 1-level containing element of the element ‘A’ .
Possible use the pham language specific code pattern element ‘([{!}] N)’ to infer some information about its neighbor element .
For example :
Code : (E B 12 (123 ([{!}] 2) C) ([{!}] 1) C K G B)
The AI system use pham language specific strict instruction to interpret this code , the result must be :
(E B 12 (123 (3 1) C) (4) C K G B)
In the result element ‘(E B 12 (123 (3 1) C) (4) C K G B)’ , note about the position of the 2 element ‘C’ . The first element ‘C’ stay next right to the element ‘(3 1)’ , which is the 2-deep region inherent index of the element ‘(3 1)’ too . The second element ‘C’ stay next right to the element ‘(4)’ , which is the local inherent index of the element ‘(4)’ too . Thus the 2-deep region inherent index of the first element ‘C’ is ‘(3 2)’ , and the local inherent index of the second element ‘C’ is ‘(5)’ .

3.1.0.8/ Pham language codepage rule repository by default contain the predefined pham language specific relative-universal-inherent-index-of-element-replacement rule :
The pham language specific code pattern element : ([{!}] […])
Where the element ‘[{!}]’ is pham language specific keyword ; the element ‘[…]’ is pham language specific keyword .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 2 nested element at 1-level of its own hierarchy structure : ‘[{!}]’ , ‘[…]’ with space between them .
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository always contain this predefined rule (the pham language specific relative-universal-inherent-index-of-element-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘([{!}] […])’ .
The AI system interpret this specific code pattern element ‘([{!}] […])’ by the pham language specific (relative-universal-inherent-index-of-element-replacement) rule :
The AI system apply this relative-universal-inherent-index-of-element-replacement rule only for element , which is in the form of the pham language specific code pattern '([{!}] [...])' .
When the AI system meet this pham language specific code pattern element ‘([{!}] […])’ in the pham language codepage , then the AI system replace this element ‘([{!}] […])’ by the inherent index of this element ‘([{!}] […])’ inside the current updated pham language codepage element .

Relative universal inherent index of element is always in the form (I{1} I{2} .. I{k}) , where I{1} , I{2} , .. , I{k} are integer . Relative universal inherent index of element use 0-based indexing model .
Feature : In each updated pham language codepage element , each element have its unique relative universal inherent index .
Feature : when AI system add small pham language codepage element (pham language message) to 1 big another pham language codepage element of a chat session , then the relative universal inherent index of all element , which are contained in this old small pham language codepage (pham language message) , will also be changed to the corresponding new pham language codepage of chat session.

For example :
Code in the pham language message :
   (something1 something2 (E B 12 (123 ([{!}] […]) C) ([{!}] […]) C K G B) something3)
In the above example , the first element '([{!}] […])' have its relative universal inherent index '(2 3 1)' , the second element '([{}] 1)' have its relative universal inherent index '(2 4)' .
Interpreting result must be :
   (something1 something2 (E B 12 (123 (2 3 1) C) (2 4) C K G B) something3)

Example use case : use case of the pham language specific code pattern element ([{!}] […])
Pham language do not use text regular expression to declare function , but it prefer to use pham language code pattern element in its 1 universal syntax , to declare function . Thus , in pham language codepage , it is impossible to directly explicitly write function to get the inherent index of the element 'A' without changing the structure of the 1-deep containing element of the element 'A' .
Possible use the pham language specific code pattern element '([{!}] […])' to infer some information about its neighbor element .
For example :
Code : Code in the pham language message :
   (something1 something2 (E B 12 (123 ([{!}] […]) C) ([{!}] […]) C K G B) something3)

The AI system use pham language specific strict instruction to interpret this code , the result must be :
    (something1 something2 (E B 12 (123 (2 3 1) C) (2 4) C K G B) something3)
In the result , note about the position of the 2 element 'C' . The first element 'C' stay next right to the element '(2 3 1)' , which is the relative universal inherent index of the element '(2 3 1)' too . The second element 'C' stay next right to the element '(2 4)' , which is the relative universal inherent index of the element '(2 4)' too . Thus the relative universal inherent index of the first element 'C' is '(2 3 2)' , and the relative universal inherent index of the second element 'C' is '(2 5)' .

3.1.0.9/ Theorem about the relation between inherent index of the containing element with inherent index of its nested element :

Given an arbitrary containing element ‘A’ , which contain other nested element .
Call the IA = (I{1} I{2} .. I{n}) as the relative universal inherent index of the element ‘A’ , where I{1} , I{2} , .. , I{n} are integer .
The following theorem is true : all contained element of the element ‘A’ must have relative universal inherent index (I{1} I{2} .. I{n} I{n+1} I{n+2} .. I{m}) . It mean that first {n} nested element of the relative universal inherent index of the contained element of the element ‘A’ must be I{1} , I{2} , .. , I{n} .
If IA is not the relative universal inherent index of the element ‘A’ , but is the inherent index of the element ‘A’ in specified element ‘BigElement’ , then there is analogous confirmation : all contained element of the element ‘A’ must have their inherent index inside the element ‘BigElement’ as (I{1} I{2} .. I{n} I{n+1} I{n+2} .. I{m}) . It mean that first {n} nested element of the inherent index of the contained element of the element ‘A’ must be I{1} , I{2} , .. , I{n} .
Thus the AI system can use the relative universal inherent index of element to fast check the whether an element is contained inside another element .

3.1.0.10/ Theorem about invariant of inherent index of element when apply pham language specific replacement rule :
When the pham language codepage are continuously updated by only the pham language specific rule , then the inherent index of an arbitrary element ‘A’ in updated pham language codepage is unchanged .
It is fundamental core theorem of pham language architecture . All pham language specific rule and instruction in this version were carefully designed to commit this theorem . All new pham language specific rule in future version will also must be designed , so that they commit this fundamental theorem .
This fundamental theorem allow many efficient and flexible way to represent pham language codepage in flexible model of other low-level programming language .

Because the pham language specific replacement rule strictly keep the principle : replace only 1 in-line met element by 1 element , always 1 by 1 , but not 1 by 2 , but not 1 by 0 . Thus the pham language specific replacement rule can annihilate element ‘A’ (make element ‘A’ to be disappeared in pham language code page) when this element ‘A’ is contained inside a replaced element , but pham language specific replacement rule do not change inherent index of element ‘A’ when the element ‘A’ do not disappear .
This theorem allow the AI system to more effectively manage and update inherent index of all element when update pham language codepage and implement pham language specific replacement rule .

3.1.0.11/ pham language specific conceptual logical executing order of element in pham language codepage :
The pham language specific conceptual logical executing order of all element in pham language codepage is SERIAL , by to execute (replacement - interpret) element with lower pham language specific logic executing order firstly , then to execute (replacement – interpret) element with higher pham language specific logic executing order .

The pham language specific conceptual logical executing order of all element in the pham language codepage is applied only in the first stage of interpreting when apply only pham language specific replacement rule . After that stage , the original pham language codepage already evolve into concrete conceptual pham language codepage , which is concrete intermediate conceptual interpretation of the original pham language codepage . After that , the AI system will use AI system specific knowledge to interpret this intermediate conceptual interpretation of original pham language codepage into final interpretation of pham language codepage .
For example :
Code : (E1 E2 (E3 (E4 E5 E1) E2))
In the above example , the pham language specific logic executing order of element are as follow :
'(E1 E2 (E3 (E4 E5 E1) E2))' , 'E1' , 'E2' , '(E3 (E4 E5 E1) E2)' , 'E3' , '(E4 E5 E1)' , 'E4' , 'E5' , 'E1' , 'E2' .

3.1.1/ Pham language codepage rule repository by default contain the pham language specific absolute-identical-equality-comparison rule :
When AI system load the pham language code page (or other used term 'pham language command' , 'pham language message'), the AI system receive full information (full contain) of each element of the loaded pham language code page, and the strict hierarchy structure of the loaded pham language code page, and all own hierarchy structure of all element of the loaded pham language code page.
Among all element of the loaded pham language code page, there are some special type of element, which are called pham language specific code pattern element, or shortly specific code pattern element. Pham language specific code pattern element are also written in the universal syntax of pham language, as any other pham language element, too. Pham language specific code pattern element are used to describe some pham language specific interpreting rule for AI system.
According to the 'Definite Number Of Element Principle Of Pham language' (refer to part 2.4.2/) , the loaded pham language codepage can be updated into new updated pham language codepage , but at any moment , the number of all element of the updated pham language codepage are always different finite number .
Pham language will normalize all element, as describe in part 2 , to deal with unnecessary space .
Because element of the pham language code page are written in text file (or in text message in the process of calculating), thus the writing-element-description-code of each element are simply text string (or hypertext string) . Thus if writing-element-description-code of the one element is the same as the writing-element-description-code of the other element in the updated pham language codepage, then the AI system must reckon that this one element and this other element are pham language specific absolute identical equal , or shortly absolute identical equal .
The pham language specific absolute-identical-equality-comparison rule state that : For 2 arbitrary pham language element , if writing-element-description-code of the first one element is the same as writing-element-description-code of the second element , then AI system must with-highest-priority reckon that these 2 element are pham language specific absolute identical equal , or shortly absolute identical equal , in the scope of the updated pham language codepage .

Absolute identical equality is pham language specific concept , which will used dedicatedly for pham language specific strict instruction . Absolute identical equality between 2 element do not confirm anything about the AI system concept of equality between these 2 element .

3.1.2/ Pham language specific strict instruction for interpreting : pham language specific (universal-element-replacement) interpreting rule . There are 2 rule :

The predefined pham language specific (universal-element-replacement rule-declaring) rule , which is used to add new rule to the pham language codepage rule repository .
And the user-defined (universal-element-replacement(Element1 , Element2)) rule , which is added to the pham language codepage rule repository rule when AI system implement the predefined (universal-element-replacement rule-declaring) rule , is used to instruct how to replace other arbitrary element , which is absolute identical equal to the element 'Element1' .

3.1.2.1/ Pham language codepage rule repository by default contain the predefined pham language specific universal-element-replacement rule-declaring rule :

The pham language specific code pattern : (Element1 [=] Element2)
Where element ‘Element1’ , element ‘Element2’ are arbitrary element ; element ‘[=]’ is pham language specific keyword element .
These specific code pattern element ‘(Element1 [=] Element2)’ always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 3 nested element at 1-level of its own hierarchy structure : ‘Element1’ , ‘[=]‘ , ‘Element2’ , with space between them .
Now given the element ‘Element1’ and given the element ‘Element2’ .
The pham language specific code pattern element : (Element1 [=] Element2)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific universal-element-replacement rule-declaring) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern rule-declaring element ‘(Element1 [=] Element2)’ .
The AI system apply this universal-element-replacement rule-declaring rule only for element , which is in form of the pham language specific code pattern '(Element1 [=] Element2)' .
When the AI system meet such pham language code pattern element , then the AI system do not update the element ‘Element1’ and ‘Element2’ by applying other pham language specific replacement rule . The AI system simply keep the element ‘(Element1 [=] Element2)’ unchanged , and its nested element ‘Element1’ , ‘Element2’ unchanged in the pham language codepage .
First , the AI system define the local inherent index of the element ‘(Element1 [=] Element2)’ . If necessary , than the AI system can decide to find the relative universal inherent index of this rule-declaring element ‘(Element1 [=] Element2)’ .
Rule declaring event inherent index : it is the inherent index of the pham language specific code pattern rule-declaring element . Thus , there are : rule declaring event local inherent index , rule declaring event {n}-deep region inherent index , rule declaring event relative universal inherent index . In this case of the universal element replacement rule ,

which will be added to the pham language codepage rule repository , they are local inherent index of the element ‘(Element1 [=] Element2)’ , {n}-deep region inherent index of the element ‘(Element1 [=] Element2)’ , relative universal inherent index of the element ‘(Element1 [=] Element2)’ . The most important are the rule declaring event local inherent index , and the rule declaring event relative universal inherent index . The AI system can also save few rule declaring event inherent index together with rule to better specify rule applying scope .
The replacement condition of rule : they are condition , when the AI system interpret some the element ‘TheElement’ , if the element ‘TheElement’ satisfy specified these condition , by which the AI system will replace the element ‘TheElement’ by other specified element .
When the AI system meet the pham language code pattern element ‘(Element1 [=] Element2)’ , then the AI system will add a new rule (universal-element-replacement(Element1 , Element2) rule) together with the rule applying scope to the pham language codepage rule repository .
Preliminary rule description : The rule (universal-element-replacement(Element1 , Element2)) mean that from this moment when the AI system interpret any other element ‘TheElement’ , which is absolute identical equal to the element ‘Element1’ , then the AI system must replace this element ‘TheElement’ by the element ‘Element2’ . The replacement condition of this rule is that : the input element must be absolute identical equal to the element ‘Element1’ . Detailed rule description : will be on next part . Rule applying scope : the (universal-element-replacement(Element1 , Element2)) rule is applied only for element , which are in this rule applying scope of this rule . The rule applying scope of this rule is the set of all element , which are contained in the 1-level containing element of the element ‘(Element1 [=] Element2)’ and are higher logical executing order than the element ‘(Element1 [=] Element2)’ . The AI system can apply 2 theorem (3.1.0.9/) and theorem (3.1.0.10/) to use the relative universal inherent index of element to check whether interested element is in the rule applying scope of rule (universal-element-replacement(Element1 , Element2)) . The AI system can also use 2 theorem (3.1.0.9/) and theorem (3.1.0.10/) , and use the inherent index of the element ‘(Element1 [=] Element2)’ to specify and record the rule applying scope of this rule (universal-element-replacement(Element1 , Element2)) in the pham language codepage rule repository .

*** Important note :
Result of applying the predefined (universal-element-replacement rule-declaring rule) on the pham language specific code pattern element '(Element1 [=] Element2)' is that : the AI system do not replace the element '(Element1 [=] Element2)' and all its contained element , but leave them unchanged in the pham language codepage , while the AI system only add new rule (universal-element-replacement(Element1 , Element2)) with its rule applying scope to the pham language codepage rule repository .
Note that this concrete rule universal-element-replacement(Element1 , Element2) depend on 2 parameter : 2 element : the element ‘Element1’ , the element ‘Element2’) .
*** End important note .

3.1.2.2/ Pham language specific (universal-element-replacement(Element1 , Element2) rule :

Precaution : It is not an element assignment operation , also not temporary element storing in other memory . It is simply a rule to directly replace element by other specified element in pham language codepage .

The pham language codepage rule repository by default do not contain the rule (universal-element-replacement(Element1 , Element2) rule .

But If the AI system met the pham language specific universal-element-replacement rule-declaring element '(Element1 [=] Element2)' before , thus the AI system follow the instruction of the (universal-element-replacement rule-declaring) rule and thus already added a concrete universal-element-replacement(Element1 , Element2) rule , which is corresponding to the instruction of the rule-declaring element '(Element1 [=] Element2)' , to the pham language codepage rule repository .

After that , when interpret arbitrary element , the AI system must apply this saved concrete universal-element-replacement(Element1 , Element2) rule .

Suppose that the AI system already added specific universal universal-element-replacement(otherElement1 , otherElement2) rule and together with rule applying scope to the pham language codepage rule repository before by interpreting the pham language specific code pattern (rule-declaring) element : (Element1 [=] Element2) .

When AI system meet the element 'TheElement' , which is absolute identical equal to the element 'Element1' , and is in the rule applying scope of this rule (universal-element-replacement(Element1 , Element2)) , then the AI system will do :

Case 1 : if the element 'TheElement' is pham language specific keyword element , then this rule will not replace the element 'TheElement' .

Case 2 : if the element 'TheElement' is predefined pham language specific code pattern element , then this rule will not replace the element 'TheElement' .

Case 3 : if the element 'TheElement' is the contained element inside the arbitrary other pham language code pattern rule-declaring element '(SomeElement [=] OtherElement)' , where 'SomeElement' and 'OtherElement' are arbitrary element , then this rule will not replace the element 'TheElement' . If the element 'TheElement' is the contained element inside the arbitrary pham language code pattern rule-declaring element '(([] FunctionName Variable1 Variable2 .. VariableN)) [:=] G)' , then this rule will not replace the element 'TheElement' .

Case 4 : If the element 'TheElement' is in rule applying scope of (saved universal-element-replacement(AA{1} , BB{1}) rule{1}) and also in rule applying scope of (saved universal-element-replacement(AA{2} , BB{2}) rule{2}) , .. , and also in rule applying scope of (saved universal-element-replacement(AA{n} , BB{n}) rule{n}) . And the element 'TheElement' satisfy all replacement condition of all these (saved universal-element-replacement(AA{1} , BB{1}) rule{1}) , (saved universal-element-replacement(AA{2} , BB{2}) rule{2}) , .. , (saved universal-element-replacement(AA{n} , BB{n}) rule{n}) , then

In this case , among these (saved universal-element-replacement(AA{1} , BB{1}) rule{1}) , (saved universal-element-replacement(AA{2} , BB{2}) rule{2}) , .. , (saved universal-element-replacement(AA{n} , BB{n}) rule{n}) , the AI system will apply only 1 (saved universal-element-replacement(AA{k} , BB{k}) rule{k}) , whose rule declaring event relative

universal inherent index is highest logical executing order , to the element ‘TheElement’ .
Case5 : in all other case , except (Case 1) , (Case 2) , (Case 3) , (Case 4) , then the AI system will replace the element ‘TheElement’ by the element ‘Element2’ .

For example :
Code : (A [=] 33) (AA A (B A))
Result must be: (A [=] 33) (AA 33 (B 33))
For example :
Code : (A [=] 1) (B (A)) (A [=] 2) (A + 2)
Result must be : (A [=] 1) (B (1)) (A [=] 2) (2 + 2)
For example :
Code : (B [=] KL) (A [=] (B + C)) (B A)
Result must be : (B [=] KL) (A [=] (B + C)) (KL (B + C))
For example :
Code : (A [=] 2) ((A + 3) [=] (B + C)) (A (A + 3))
Result must be : (A [=] 2) ((A + 3) [=] (B + C)) (2 (B + C))
For example :
Code : ((A [=] 1) B A) A
Result must be : ((A [=] 1) B 1) A
For example :
Code : (A [=] KK) A (A [=] LL) A
Result must be : (A [=] KK) KK (A [=] LL) LL
Code : (A [=] PP) A ((A [=] QQ) A) A
Result must be : (A [=] PP) PP ((A [=] QQ) QQ) PP

3.1.2.3/ Pham language codepage rule repository by default contain the predefined Pham language specific universal-element-replacement-in-scope rule-declaring rule :
Precaution : It is not an element assignment operation , also not temporary element storing in other memory . It is simply a rule to add new rule to the pham language codepage rule repository .
The pham language specific code pattern : (Element1 [=] Element2 (M N))
Where element ‘[=]’ is pham language specific keyword element ; ‘Element1’ , ‘Element2’ are arbitrary element ; element ‘M’ and element ‘N’ are arbitrary element .
These pham languae specific code pattern element ‘(Element1 [=] Element2 (M N))’ always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 4 1-level nested element : ‘Element1’ , '[=]' , ‘Element2’ , ‘(M N)’ with space between them . The nested element ‘(M N)’ always begin with parenthese ‘(‘ and end with parenthese ‘)’ , and have strictly 2 1-level nested element ‘M’ and ‘N’ with space between them .
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific universal-element-replacement-in-scope rule-declaring rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern rule-declaring element ‘(Element1 [=] Element2 (M N))’ .

The AI system apply the universal-element-replacement-in-scope rule-declaring rule only for element , which is in form of the pham language specific code pattern '(Element1 [=] Element2 (M N))' :
Now given the element 'Element1' and element 'Element2' and element 'M' and element 'N' .
The pham language specific code pattern element : (Element1 [=] Element2 (M N))
When the AI system meet such pham language code pattern element , then the AI system do not update and do not interpret the element 'Element1' and 'Element2' by applying other pham language specific replacement rule . But the AI system update the element 'M' and element 'N' by applying other pham language replacement rule , as a result : the element 'M' -> the element 'updatedM' , and the element 'N' -> the element 'updatedN' .
If the element 'updatedM' is not natural integer greater than 0 , or the element 'updatedN' is not natural integer greater than 0 , then the AI system replace the element '(Element1 [=] Element2 (M N))' by the element '()' , and the AI system do not add new rule to the pham language codepage rule repository .
If the element 'updatedM' is natural integer greater than 0 , and the element 'updatedN' is natural integer greater than 0 , and the AI system can not define the {updatedM}-level containing element of the element '(Element1 [=] Element2 (M N))' , then the AI system take the updated pham language codepage element as the temporary {updatedM}-level containing element of the element '(Element1 [=] Element2 (M N))' only for this case .
If the element 'updatedM' is natural integer greater than 0 , and the element 'updatedN' is natural integer greater than 0 , and the AI system must find the {updatedM}-level containing element of the element '(Element1 [=] Element2 (M N))' , then the AI replace the element '(Element1 [=] Element2 (M N))' by the element '(Element1 [=] Element2 (updatedM updatedN))' , and the AI system add a new rule (universal-element-replacement(Element1 , Element2) rule) together with rule applying scope , which is the set of all contained element of the {updatedM}-level containing element of the element '(Element1 [=] Element2 (M N))' and in the level from 1 to {updatedM + updatedN} in the the own hierarchy structure of the {updatedM}-level containing element of the element '(Element1 [=] Element2 (M N))' .
Rule declaring event inherent index of this added rule (universal-element-replacement(Element1 , Element2) rule) is the inherent index of the element '(Element1 [=] Element2 (M N))' . Thus there are rule declaring event local inherent index of this rule(universal-element-replacement(Element1 , Element2) rule) , rule declaring event {n}-deep region inherent index of this rule (universal-element-replacement(Element1 , Element2) rule) , rule declaring event relative universal inherent index of this rule (universal-element-replacement(Element1 , Element2) rule) .

For example :
Code : ((1 (A [=] 2) A) A (1 (A [=] 3 (1 3))) A)
Result must be : ((1 (A [=] 2) 2) A (1 (A [=] 3 (2 3))) 3)

3.1.3/ The pham language codepage rule repository by default contain the pham language specific inline-replacement rule :

The pham language specific code pattern : (AA [|] XX [->] YY)
Where the element ‘AA’ and the element ‘XX’ and the element ‘YY’ are arbitrary element ; the element ‘[|]’ and the element [->] are pham language keyword element .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 5 nested element at 1-level of its own hierarchy structure , with space between them .
Given the element ‘AA’ and element ‘XX’ and element ‘YY’ .
The pham language specific code pattern element : (AA [|] XX [->] YY)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific inline-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘(AA [|] XX [->] YY)’ .
The AI system apply the inline-replacement rule only for element , which is in the form of the pham language specific code pattern '(AA [|] XX [->] YY)' .

When the AI system meet the such pham language code pattern element ‘(AA [|] XX [->] YY)’ in the current pham language codepage , then the AI system firstly not update and not interpret the element ‘AA’ , element 'XX' and element 'YY' , then the AI system replace with highest priority all 'XX' inside the element 'AA' by the element 'YY' , as a result element 'AA' become element 'updatedAA' . Then the AI system replace the element ‘(AA [|] XX [->] YY)’ by the element 'updatedAA' .
Thus the AI system update the pham language codepage .
If the element ‘XX’ is pham language specific keyword element , then this rule will not replace the element ‘XX’ .
If the element ‘XX’ is pham language specific code pattern element , then this rule will not replace the element ‘XX’ .
If the element ‘XX’ is the contained element (at any level) inside the pham language specific code pattern rule-declaring element ‘(Element1 [=] Element2) , then this rule will not replace the element ‘XX’ .
If the element 'XX' is the contained element inside the pham language code pattern rule-declaring element '(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)' , then this rule will not replace the element ‘XX’ .

Thus the AI system update the pham language codepage .
For example :
Code : ((A + B (13 C + D) (1 (1 A)) (A (A (C A)))) [|] A [->] 555)
Result must be : (555 + B (13 C + D) (1 (1 555)) (555 (555 (C 555))))

3.1.4/ Pham language specific strict instruction for function declaring and function calling :
Pham language specific instruction allow to declare function , and after declaring function then possible to call function .
There is predefined pham language specific function-multi-replacement rule-declaring rule , which instruct add new rule to the pham language codepage rule repository .

There are pham language specific function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}) , G) rule , which are used to instruct how to replace element in the pham language codepage .

3.1.4.1/ Pham language codepage rule repository by default contain the predefined pham language specific function-multi-replacement rule-declaring rule :

Precaution : It is not an element assignment operation , also not temporary element storing in other memory .. It is simply a rule to instruct to add new rule to the pham language codepage rule repository .

The pham language specific code pattern :

(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)

Where the element '[]' , '[:=]' are the pham language keyword element ; the element 'FunctionName' , 'Variable1' , 'Variable2' , .. , 'VariableN' are arbitrary element ; the element 'G' is arbitrary element .

These specific code pattern element always begin with parenthese '(' and end with parenthese ')' , always have strictly 3 1-level nested element , with space between them :

'([] FunctionName Variable1 Variable2 .. VariableN)' , '[:=]' , 'G' .

The first 1-level nested element '([] FunctionName Variable{1} Variable{2} .. Variable{N})' must begin with parenthese '(' and end with parenthese ')' , must have the first 1-level nested element '[]' , must have second 1-level nested element .

Given the element 'FunctionName' and element 'Variable{1}' and element 'Variable{2}' , .. , 'Variable{n}' .

The pham language specific code pattern function-declaring element :

(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)

*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific function-multi-replacement rule-declaring rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern rule-declaring element '(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)' .

The AI system apply the rule (function-multi-replacement rule-declaring) only for element , which is in the form of pham language code pattern '(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)' .

When the AI system meet such pham language code pattern element , then the AI system do not update any element (at any level in the own hierarchy structure of the element '(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)') inside the element '(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)' by applying other pham language specific replacement rule .

First , the AI system find the local inherent index of the element '(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)' . If necessary , than the AI system can decide to find the relative universal inherent index of this rule-declaring element '(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)' to use as basic argument to define and record the rule applying scope .

Rule declaring event inherent index : it is the inherent index of the pham language specific code pattern rule-declaring element . Thus , there are : rule declaring event local inherent index , rule declaring event {n}-deep region inherent index , rule declaring event relative universal inherent index . In this case , they are local inherent index of the element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)’, {n}-deep region inherent index of the element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)’ , relative universal inherent index of the element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)’ . The most important are the rule declaring event local inherent index , and the rule declaring event relative universal inherent index . The AI system can also save few rule declaring event inherent index together with rule to better specify rule applying scope .
The replacement condition of rule : they are condition , when the AI system interpret some the element ‘TheElement’ , if the element ‘TheElement’ satisfy specified these condition , by which the AI system will replace the element ‘TheElement’ by other specified element .
When the AI system meet the pham language code pattern rule-declaring element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)’ , then the AI system will add a new rule (function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)) together with the rule applying scope to the pham language codepage rule repository . Preliminary rule description : The rule (function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)) mean that from this moment when the AI system meet any other pham language specific code pattern element ‘([] FName Value{1} Value{2} .. Value{n})’ , whose first 1-level nested element is ‘[]’ , and whose second 1-level nested element ‘FName’ is absolute identical equal to the element ‘FunctionName’ , then the AI system must replace this element ‘([] FName Value{1} Value{2} .. Value{n})’ by the element ‘G’ , and immediate after that the AI system must apply consequently all {N} in-line replacement (Variable{1} -> Value{1} , Variable{2} -> Value{2} , .. , Variable{n} -> Value{n}) for all element inside the element G , so that element 'G' become element 'updatedG' , and as a result the AI system replace the original pham language code pattern element ‘([] FName Value{1} Value{2} .. Value{n})’ by the element ‘updatedG’ .
The replacement condition of this rule is that : the input element must be pham language specific code pattern element ‘([] FName Value{1} Value{2} .. Value{N})’ , whose first 1-level nested element ‘[]’ , and whose second 1-level nested element ‘FName’ is absolute identical equal to the element ‘FunctionName’ . Detailed rule description : will be on next part . Rule applying scope : this rule is applied only for element , which are in this rule applying scope of this rule . The rule applying scope of this rule is the set of all element , which are contained in the 1-level containing element of the rule-declaring element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)’ and are higher logical executing order than the element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)’ . The AI system can apply 2 theorem (3.1.0.9/) and theorem (3.1.0.10/) to use the relative universal inherent index of the arbitrary interested element to check whether the interested element is in the rule applying scope of this rule . The AI system can also use 2 theorem (3.1.0.9/) and theorem

(3.1.0.10/) , and use the rule declaring event inherent index to specify and record the rule applying scope of this rule (function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)) in the pham language codepage rule repository .
** Result of applying the predefined (function-multi-replacement rule-declaring) rule on the pham language specific code pattern rule-declaring element '(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)' is that : the AI system do not replace the element '(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)' and all its contained element , but leave them unchanged in the pham language codepage , while the AI system only add new rule (function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)) with its rule applying scope to the pham language codepage rule repository .

*** Important note :
Result of applying the predefined (function-multi-replacement rule-declaring rule) on the pham language specific code pattern element '(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)' is that : the AI system do not replace the element '(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)' and all its contained element , but leave them unchanged in the pham language codepage , while the AI system only add new concrete rule (function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)) together with its rule applying scope to the pham language codepage rule repository.
And note that this new concrete rule (function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)) depend on the following 2 parameter : the following 2 element : the element ‘(FunctionName Variable{1} Variable{2} .. Variable{n})’ and the element ‘G’ . While the first parameter : the element ‘(FunctionName Variable{1} Variable{2} .. Variable{n})’ depend on other element : the element ‘FunctionName’ , the element ‘Variable{1}’ , the element ‘Variable{2} , .. , the element ‘Variable{n}’ .
*** End important note .

3.1.4.2/ Pham language specific (function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)) rule :
Precaution : It is not an element assignment operation , also not temporary element storing in other memory . It is simply a rule to directly replace element by other specified element in pham language codepage .

The pham language codepage rule repository by default do not contain the rule (function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)) .
But If the AI system did meet the pham language specific function-multi-replacement rule-declaring element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)’ in the pham language codepage before , thus the AI system followed the instruction of the (function-multi-replacement rule-declaring) rule and thus already added a concrete rule (function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)), which is corresponding to the instruction of the rule-declaring element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)’ , to the pham language codepage rule repository .

After that , when interpret arbitrary interested element , the AI system must check whether the interested element is in the form of the pham language specific code pattern (FunctionName , Variable{1} , Variable{2} , .. , Variable{n}) , in order to apply this saved rule (function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)) .
Suppose that the AI system already added specific rule (function-multi-replacement((otherFunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)) and together with rule applying scope , to the pham language codepage rule repository before by interpreting the pham language specific code pattern (rule-declaring) element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)’.

If there were added pham language specific function-multi-replacement rule in the pham language codepage repository , then when AI system meet pham language specific code pattern element ‘([] FName Value{1} Value{2} .. Value{n})’ , the AI system must apply pham language function-multi-replacement rule to to control how to replace the met pham language specific code pattern element ‘([] FName Value{1} Value{2} .. Value{n})’ by the specified element .

Suppose that the AI system already added pham language specific function-multi-replacement rule (function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)) and together with rule applying scope in the pham language codepage rule repository before by interpreting the pham language specific code pattern (rule-declaring) element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)’ .
When AI system meet the any pham language specific code pattern element ‘([] FName Value{1} Value{2} .. Value{n})’ , then the AI system will do :
The AI system will not update the 1-level nested element ‘FName’ . If the AI system see that the 1-level nested element ‘FName’ is absolute identical equal to the element ‘FunctionName’ , then the AI system will do :
The AI system will interpret and update all element ‘Value{1}’ , ‘Value{2}’ , .. , ‘Value{n}’ by other pham language specific replacement rule , as a result : the element ‘Value{1}’ -> the element ‘updatedValue{1}’ , the element ‘Value{2}’ -> the element ‘updatedValue{2}’ , .. , the element ‘Value{n}’ -> the element ‘updatedValue{n}’ .
Case 1 : if the element ‘FName’ is pham language specific keyword element , then this rule will not replace the element ‘([] FName Value{1} Value{2} .. Value{n})’ .
Case 2 : if the element ‘FName’ is predefined pham language specific code pattern element , then this rule will not replace the element ‘([] FName Value{1} Value{2} .. Value{n})’ .
Case 3 : If the element ‘([] FName Value{1} Value{2} .. Value{n})’ is in rule applying scope of (saved (function-multi-replacement((FunctionName{1} , Variable{1} , Variable{2} , .. , Variable{n}),G)) rule{1}) and also in rule applying scope of (function-multi-replacement((FunctionName{2} , Variable{1} , Variable{2} , .. , Variable{n}),G)) rule{2}) , .. , and also in rule applying scope of (function-multi-replacement((FunctionName{k} , Variable{1} , Variable{2} , .. , Variable{n}),G)) rule{k}) . And the element ‘([] FName Value{1}

Value{2} .. Value{n})’ satisfy all replacement condition of all these rule , (it mean that 'FName' is absolute identical equal to all following element 'FunctionName{1}' , 'FunctionName{2}' , .. , 'FunctionName{k}')
In this case , among these saved function-multi-replacement rule , the AI system will apply only 1 saved function-multi-replacement rule , whose rule declaring event relative universal inherent index is highest logical executing order , to the element ‘([] FName Value{1} Value{2} .. Value{n})’ .
Case4 : in all other case , except Case 1 , Case 2 , Case 3 , then the AI system will replace the met pham language specific code pattern element ‘([] FName Value{1} Value{2} .. Value{n})’ by the element ‘G’ , and immediately the AI system keep delay update the element G , and then the AI system consequently apply {N} inline-replacement (Variable{1} -> updatedValue{1} , Variable{2} -> updatedValue{2} , .. , Variable{n} -> updatedValue{n}) for all element inside the element ‘G’ , as a result the AI system update the element ‘G’ into the element ‘updatedG’ , and then the AI system can update the element ‘updatedG’ into the element ‘nextUpdatedG’ , and finally the AI system replace the original pham language code pattern element ‘([] FName Value{1} Value{2} .. Value{n})’ by the element ‘nextUpdatedG’ .
For example :
Code :

```
          (A (([] myFunction x y z) [:=] (x + y + z)) ([] myFunction 1 2 3) ABC)
```

Result must be : (A (([] myFunction x y z) [:=] (x + y + z)) (1 + 2 + 3) ABC)
For example :
Code : ((x [=] 3) (([] myFunction x y z) [:=] (x 11 y 11 z)) x ([] myFunction A B C) x)
Result must be : ((x [=] 3) (([] myFunction x y z) [:=] (x 11 y 11 z)) 3 (A 11 B 11 C) 3)

3.1.4.3/ The Pham language codepage rule repository by default contain the predefined pham language specific function-multi-replacement-in-scope rule-declaring rule :
Precaution : It is not an element assignment operation , also not temporary element storing in other memory . It is simply a rule to instruct how to add new rule to the pham language codepage rule repository .
The pham language specific code pattern :

```
     (([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G (M P))
```

Where the element ‘[]’ , ‘[:=]’ are the pham language keyword element ; the element ‘FunctionName’ , ‘Variable{1}’ , ‘Variable{2}’ , .. , ‘Variable{n}’ are arbitrary element ; the element ‘G’ is arbitrary element ; element ‘M’ and ‘P’ are arbitrary element .
These pham language specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 4 1-level nested element, with space between them , :
‘([] FunctionName Variable{1} Variable{2} .. Variable{n})’ , ‘[:=]’ , ‘G’ , ‘(M P)’
The first 1-level nested element ‘([] FunctionName Variable1 Variable2 .. VariableN)’ must begin with parenthese ‘(‘ and end with parenthese ‘)’ ,

must have the first 1-level nested element ‘[]’ , must have second 1-level nested element .
The nested element ‘(M P)’ always begin with parenthese ‘(‘ and end with parenthese ‘)’ , and have strictly 2 1-level nested element ‘M’ and ‘P’ with space between them .
Now given the element ‘FunctionName’ , ‘Variable{1}’ , ‘Variable{2}’ , .. , ‘Variable{n}’ , and element ‘M’ and element ‘N’ .
The pham language specific code pattern rule-declaring element : (([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G (M P))
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific function-multi-replacement-in-scope rule-declaring rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern rule-declaring element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G (M P))’ .
The AI system apply the rule (function-multi-replacement-in-scope rule-declaring) only for element , which is in the form of the pham language specific code pattern ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G (M P))’ .

This is pham language specific rule-declaring element , whose functionality is absolute analogous to the pham language specific function-multi-replacement rule-declaring element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G)’ , with the only difference is that there is other specified rule applying scope in this case .
When the AI system meet such pham language code pattern element , then the AI system do not update all element , except the 2 element ‘M’ and ‘P’ inside the element ‘(M P)’ , inside the element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G (M P))’ , by applying other pham language replacement rule . But the AI system update the element ‘M’ and element ‘P’ by applying other pham language replacement rule , as a result : the element ‘M’ -> the element ‘updatedM’ , and the element ‘P’ -> the element ‘updatedP’ .
If the element ‘updatedM’ is not natural integer greater than 0 , or the element ‘updatedP’ is not natural integer greater than 0 , then the AI system replace the element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G (M P))’ by the element ‘()’ .
If the element ‘updatedM’ is natural integer greater than 0 , and the element ‘updatedP’ is natural integer greater than 0 , and the AI system can not define the {updatedM}-level containing element of the element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G (M P))’ , then the AI system take the updated pham language codepage element as the temporary {updatedM}-level containing element of the element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G (M P))’ only for this case .
If the element ‘updatedM’ is natural integer greater than 0 , and the element ‘updatedP’ is natural integer greater than 0 , and the AI system can define the {updatedM}-level containing element of the element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G (M P))’ , then the AI replace the element ‘(([] FunctionName Variable{1}

Variable{2} .. Variable{n}) [:=] G (M P))’ by the element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G (updatedM updatedP))’ , and the AI system add a new rule (function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)) together with new rule applying scope , which is the set of all contained element of the {updatedM}-level containing element of the element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G (M P))’ and in all level from 1 to {updatedM + updatedP} of the own hierarchy structure of the {updatedM}-level containing element of the element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G (M P))’ .
Rule declaring event inherent index of this added rule (function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)) is the inherent index of the element ‘(([] FunctionName Variable{1} Variable{2} .. Variable{n}) [:=] G (M P))’ . Thus there are rule declaring event local inherent index of this added rule (function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)), rule declaring event {n}-deep region inherent index of this added rule (function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)) , rule declaring event relative universal inherent index of this added rule (function-multi-replacement((FunctionName , Variable{1} , Variable{2} , .. , Variable{n}),G)) .
For example :
Code : (12 (([] myFunction A B) [:=] (A xx B)) ([] myFunction X Y) 11) ([] myFunction X Y)
Result must be : (12 (([] myFunction A B) [:=] (A xx B)) (X xx Y) 11) ([] myFunction X Y)
Code : (12 (([] myFunction A B) [:=] (A xx B) (2 3)) ([] myFunction X Y) 11) ([] myFunction X Y)
Result : (12 (([] myFunction A B) [:=] (A xx B) (2 3)) (X xx Y) 11) (X xx Y)

3.1.5/ The pham language codepage rule repository by default contain the predefined pham language specific absolute-identical-equality-replacement rule :
The pham language specific code pattern : (element1 [==] element2)
Where element ‘element1’ and element ‘element2’ are arbitrary element , the element ‘[==]’ is pham language keyword element .
The specific code pattern element ‘(element1 [==] element2)’ always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 3 nested element at 1-level of its own hierarchy structure : ‘element1’ , ‘[==]‘ , ‘element2’ , with space between them .
Now given the element ‘element1’ and given the element ‘element2’ .
The pham language specific code pattern element : (element1 [==] element2)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific absolute-identical-equality-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘(element1 [==] element2)’ .

The AI system apply the pham language specific absolute-identical-equality-replacement rule only for element , which is in form of the pham language specific code pattern '(element1 [==] element2)' .

When the AI system meet the such pham language code pattern element ‘(element1 [==] element2)’ in the current pham language codepage , then the AI system firstly interpret and update the element ‘element1’ and the element ‘element2’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘element1’ -> the element ‘updatedElement1’ , and the element ‘element2’ -> the element ‘updatededElement2’ . Then the AI system replace the element ‘(element1 [==] element2)’ by the element ‘[1?]’ , which is the Boolean Logic True in pham language, if the element ‘updatedElement1’ and the element ‘updatedElement2’ are absolute identical equal , and by the element ‘[0?]’ , which is the Boolean Logic False in pham language , if the element ‘updatedElement1’ and the element ‘updatedElement2’ are not absolute identical equal . Thus the AI system updated the current pham language codepage into updated pham language codepage .
The element ‘[1?]’ , and element ‘[‘0?’]’ are pham language keyword element .
For example :
Pham language code page : (this element is not equal to any element) (element2) (someElement (this element is not equal to any element))
In this example , the first element ‘(this element is not equal to any element)’ is absolute identical equal to the second element ‘(this element is not equal to any element)’ .
For example :
Pham language message : ((3 – 0) = someElement) ((3) otherElement)
In this example, in this loaded pham language code page, the element ‘(3 - 0)’ have its writing-element-description-code : “(3 - 0)”. The element ‘(3)’ have its writing-element-description-code : “(3)” . These writing-element-description-code are different , thus the pham language specific absolute identical equality interpreting rule tell that these element are not absolute identical equal. But in much many user context case, most AI system can reckon that these element ‘(3 - 0)’ and element ‘(3)’ are equal or identical equal. It is ok, and it do not conflict with pham language design principle.
For example :
Pham language message : ((3 – 0) = someElement) ((3- 0) otherElement)
In this example, in this loaded pham language code page, the element ‘(3 - 0)’ have its writing-element-description-code : “(3 - 0)”. The element ‘(3- 0)’ have its writing-element-description-code : “(3- 0)” . Note about space inside these element. These writing-element-description-code are different, thus the pham language specific absolute identical equality interpreting rule tell these element are not absolute identical equal . But in much many user context case, most AI system can reckon that these element ‘(3 - 0)’ and element ‘(3- 0)’ are equal or identical equal. It is ok, and it do not conflict with pham language design principle.

3.1.6/ The pham language codepage rule repository by default contain the predefined pham language specific equivalent-belonging-replacement rule :
The pham language specific code pattern : (elementA [<:] elementB)

Where element ‘elementA’ and element ‘elementB’ are arbitrary element , element ‘[<:]’ is pham language keyword element .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 3 nested element at 1-level of its own hierarchy structure , with space between them .
Now given the element ‘elementA’ , give the element ‘elementB’ .
There is pham language specific code pattern element : (elementA [<:] elementB)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific equivalent-belonging-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘(elementA [<:] elementB)’ .
The AI system apply the pham language specific equivalent-belonging-replacement rule only for element , which is in the form of the pham language specific code pattern '(elementA [<:] elementB)' .
The AI system interpret this specific code pattern element ‘(elementA [<:] elementB)’ by the pham language specific (equivalent-belonging) rule :
When the AI system meet the such pham language specific code pattern element ‘(elementA [<:] elementB)’ in the current pham language codepage , then the AI system firstly update the element ‘elementA’ and the element ‘elementB’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘elementA’ -> the element ‘updatedElementA’ , and the element ‘elementB’ -> the element ‘updatedElementB’ . Then the AI system replace the element ‘(elementA [<:] elementB)’ by the element ‘[?1]’ (it is pham language Boolean Logic True) if the AI system see that in the 1-level of the own hierarchy structure of the element ‘updatedElementB’ there is at least 1 nested element , which is absolute identical equal to the element ‘updatedElementA’ . The AI system replace the element ‘(elementA [<:] elementB)’ by the element [0?]’ (it is the pham language Boolean Logic False) if the AI system see that in the 1-level of the own hierarchy structure of the element ‘updatedElementB’ there is no any nested element , which is absolute identical equal to the element ‘updatedElementA’ .
Thus the AI system just already updated the pham language codepage .

3.1.6.1/ The pham language codepage rule repository by default contain the predefined pham language specific equivalent-belonging-№2-replacement rule :
The pham language specific code pattern : (elementA [<::] elementB)
Where element ‘elementA’ and element ‘elementB’ are arbitrary element , element ‘[<::]’ is pham language keyword element .
In these code pattern , the element ‘elementA’ and the element ‘elementB’ can be arbitrary element , the element ‘[<::]’ is pham language specific keyword element .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 3 nested element at 1-level of its own hierarchy structure , with space between them .
Now given the element ‘elementA’ , given the element ‘elementB’ .
There is pham language specific code pattern element : (elementA [<::] elementB)

*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific equivalent-belonging-№2-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘(elementA [<::] elementB)’ .
The AI system apply the the pham language specific equivalent-belonging-№2-replacement rule only for element , which is in the form of the pham language specific code pattern '(elementA [<::] elementB)' .
When the AI system meet the such pham language code pattern element ‘(elementA [<::] elementB)’ in the current pham language codepage , then the AI system firstly update the element ‘elementA’ and the element ‘elementB’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘elementA’ -> the element ‘updatedElementA’ , and the element ‘elementB’ -> the element ‘updatedElementB’ . Then the AI system replace the element ‘(elementA [<::] elementB)’ by the element ‘[?1?]’ (it is the pham language Boolean Logic True) if the AI system see that in the own hierarchy structure of the element ‘updatedElementB’ there is at least 1 element , which is absolute identical equal to the element ‘updatedElementA’ . The AI system replace the element ‘(elementA [<::] elementB)’ by the element ‘[?0?]’ (it is the pham language Boolean Logic False) if the AI system see that in the own hierarchy structure of the element ‘updatedElementB’ there is no any element , which is absolute identical equal to the element ‘updatedElementA’ . Thus the AI system just already updated the pham language codepage .

3.1.7/ Pham language specific strict instruction for interpreting : pham language specific triple-Boolean-arithmetic rule :
Pham language specific instruction use triple Boolean arithmetic to control conditional execution and conditional interpretation .

3.1.7.1/ The pham language codepage rule repository by default contain the predefined pham language specific triple-Boolean-value-replacement rule :
The pham language specific code pattern : ([?3?] TheElement)
Where the element ‘TheElement’ is arbitrary element ; the element ‘[?3?]’ is pham language keyword element .
In this code pattern , the element ‘TheElement’ can be arbitrary element , the element ‘[?3?]’ is pham language specific keyword element .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 2 nested element at 1-level of its own hierarchy structure , with space between them .
Given the element ‘TheElement’ .
The pham language specific code pattern element : ‘([?3?] TheElement)’
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific triple-Boolean-value-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘([?3?] TheElement)’ .

The AI system apply the pham language specific triple-Boolean-value-replacement rule only for element , which is in the form of the pham language specific code pattern ‘([?3?] TheElement)’ .
When the AI system meet the such pham language code pattern element ‘([?3?] TheElement)’ in the current pham language codepage , then the AI system firstly update the element ‘TheElement’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘TheElement’ -> the element ‘updatedTheElement’ .
If the AI system reckon that the element ‘updatedTheElement’ is a standard Boolean Logic True element , then the AI system replace the element ‘([?3?] TheElement)’ by the element ‘[1?]’ , which is pham language specific Boolean Logic True . If the AI system reckon that the element ‘updatedTheElement’ is a standard Boolean Logic False , then the AI system replace the element ‘([?3?] TheElement)’ by the element ‘[0?]’ , which is pham language specific Boolean Logic False element .
Besides , if the AI system see that the element ‘updatedTheElement’ is absolute identical equal to the element ‘[0?]’ , then the AI system must definitely replace the element ‘([?3?] TheElement)’ by the element ‘[0?]’ . If the AI system see that the element ‘updatedTheElement’ is absolute identical equal to the element ‘[1?]’ , then the AI system must definitely replace the element ‘([?3?] TheElement)’ by the element ‘[1?]’ .
In any other case , which are not the above 2 case , then the AI system replace the element ‘([?3?] TheElement)’ by the element ‘()’ .
Thus the AI system update the pham language codepage .

3.1.7.2/ The pham language codepage rule repository by default contain the predefined pham language specific triple-Boolean-value-negation-replacement rule :
The pham language specific code pattern : ([!?] theElement)
Where the element ‘theElement’ is arbitrary element ; the element ‘[!?]’ is pham language keyword element .
In this code pattern , the element ‘theElement’ can be arbitrary element , the element ‘[!?]’ is pham language specific keyword element .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 2 nested element at 1-level of its own hierarchy structure , with space between them .
Given the element ‘theElement’ .
The pham language specific code pattern element : ‘([!?] theElement)’
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific triple-Boolean-value-negation-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘([!?] theElement)’ .
The AI system apply the pham language specific triple-Boolean-value-negation-replacement rule only for element , which is in the form of the pham language specific code pattern ‘([!?] theElement)’ .
When the AI system meet the such pham language code pattern element ‘([!?] theElement)’ in the current pham language codepage , then the AI system firstly update the element ‘theElement’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘theElement’ -> the element ‘updatedTheElement’ .

If the AI system reckon that the element ‘updatedTheElement’ is a standard Boolean Logic True element, then the AI system replace the element ‘([!?] theElement)’ by the element ‘[0?]’ , which is the pham language specific Boolean Logic False . If the AI system reckon that the element ‘updatedTheElement’ is a standard Boolean Logic False element, then the AI system replace the element ‘([!?] theElement)’ by the element ‘[1?]’ , which is the pham language specific Boolean Logic True . Besides , if the AI system see that the element ‘updatedTheElement’ is absolute identical equal to the element ‘[0?]’ , then the AI system must definitely replace the element ‘([!?] theElement)’ by the element ‘[1?]’ . If the AI system see that the element ‘updatedTheElement’ is absolute identical equal to the element ‘[1?]’ , then the AI system must definitely replace the element ‘([!?] theElement)’ by the element ‘[0?]’ .
In any other case , which are not the above 2 case , then the AI system replace the element ‘([!?] theElement)’ by the element ‘()’ . Thus the AI system update the pham language codepage .

3.1.7.3/ The pham language codepage rule repository by default contain the predefined pham language specific triple-Boolean-union-replacement rule :
The pham language specific code pattern : (AA [+?] BB)
Where the element ‘AA’ and the element ‘BB’ are arbitrary element ; the element ‘[+?]’ is pham language keyword element .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 3 nested element at 1-level of its own hierarchy structure , with space between them .
Given the element ‘AA’ and element ‘BB’ .
The pham language specific code pattern element : (AA [+?] BB)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific triple-Boolean-union-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘(AA [+?] BB)’ .
The AI system apply the pham language specific triple-Boolean-union-replacement rule only for element , which is in the form of the pham language specific code pattern '(AA [+?] BB)' .
The AI system interpret this pham language specific code pattern element ‘(AA [+?] BB)’ by the pham language specific (triple-Boolean-union-replacement) rule :
When the AI system meet the such pham language code pattern element ‘(AA [+?] BB)’ in the current pham language codepage , then the AI system firstly update the element ‘AA’ and the element ‘BB’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘AA’ -> the element ‘updatedAA’ , the element ‘BB’ -> the element ‘updatedBB’ . If the AI system see that the element ‘updatedAA’ is not Boolean Logic Value ‘(0?)’ or ‘(1?)’ , or the element ‘updatedBB’ is not Boolean Logic Value ‘(0?)’ or ‘(1?)’ , then the AI system replace the element ‘(AA [+?] BB)’ by the element ‘()’ .
If the AI system see that the element ‘updatedAA’ is a Boolean Logic Value ‘(0?)’ or ‘(1?)’ , and the element ‘updatedBB’ is Boolean Logic Value ‘(0?)’ or ‘(1?)’ , then the AI system replace the element ‘(AA

[+?] BB)’ by the result of standard Boolean Logic Union operation between 2 Boolean Logic Value ‘updatedAA’ and ‘updatedBB’ .
Thus the AI system update the pham language codepage .

3.1.7.4/ The pham language codepage rule repository by default contain the predefined pham language specific triple-Boolean-distraction-replacement rule :
The pham language specific code pattern : (AA [-?] BB)
Where the element ‘AA’ and the element ‘BB’ are arbitrary element ; the element ‘[-?]’ is pham language keyword element .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 3 nested element at 1-level of its own hierarchy structure , with space between them .
Given the element ‘AA’ and element ‘BB’ .
The pham language specific code pattern element : (AA [-?] BB)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific triple-Boolean-distraction-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘(AA [-?] BB)’ .
The AI system apply the pham language specific triple-Boolean-distraction-replacement rule only for element , which is in the form of the pham language specific code pattern '(AA [-?] BB)' .
When the AI system meet the such pham language code pattern element ‘(AA [-?] BB)’ in the current pham language codepage , then the AI system firstly update the element ‘AA’ and the element ‘BB’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘AA’ -> the element ‘updatedAA’ , the element ‘BB’ -> the element ‘updatedBB’ . If the AI system see that the element ‘updatedAA’ is not Boolean Logic Value ‘(0?)’ or ‘(1?)’ , or the element ‘updatedBB’ is not Boolean Logic Value ‘(0?)’ or ‘(1?)’ , then the AI system replace the element ‘(AA [-?] BB)’ by the element ‘()’ .
If the AI system see that the element ‘updatedAA’ is a Boolean Logic Value ‘(0?)’ or ‘(1?)’ , and the element ‘updatedBB’ is Boolean Logic Value element ‘(0?)’ or ‘(1?)’, then the AI system replace the element ‘(AA [-?] BB)’ by the result of standard Boolean Logic Distraction operation between 2 Boolean Logic Value ‘updatedAA’ and ‘updatedBB’ .
Thus the AI system update the pham language codepage .

3.1.7.5/ The pham language codepage rule repository by default contain the predefined pham language specific triple-Boolean-intersection-replacement rule :
The pham language specific code pattern : (AA [&?] BB)
Where the element ‘AA’ and the element ‘BB’ are arbitrary element ; the element ‘[&?]’ is pham language keyword element .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 3 nested element at 1-level of its own hierarchy structure , with space between them .
Given the element ‘AA’ and element ‘BB’ .
The pham language specific code pattern element : (AA [&?] BB)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage

rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific triple-Boolean-intersection-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element '(AA [&?] BB)' .
The AI system apply the pham language specific triple-Boolean-intersection-replacement rule only for element , which is in the form of the pham language specific code pattern '(AA [&?] BB)' .
The AI system interpret this pham language specific code pattern element '(AA [&?] BB)' by the pham language specific (triple-Boolean-intersection) rule :
When the AI system meet the such pham language code pattern element '(AA [&?] BB)' in the current pham language codepage , then the AI system firstly update the element 'AA' and the element 'BB' by applying other pham language specific rule (pham language specific replacement rule) to update : the element 'AA' -> the element 'updatedAA' , the element 'BB' -> the element 'updatedBB' . If the AI system see that the element 'updatedAA' is not Boolean Logic Value '(0?)' or '(1?)' , or the element 'updatedBB' is not Boolean Logic Value '(0?)' or '(1?)' , then the AI system replace the element '(AA [&?] BB)' by the element '()' .
If the AI system see that the element 'updatedAA' is a Boolean Logic Value '(0?)' or '(1?)', and the element 'updatedBB' is Boolean Logic Value '(0?)' or '(1?)' , then the AI system replace the element '(AA [&?] BB)' by the result of standard Boolean-Logic-Intersection operation between 2 Boolean Logic Value 'updatedAA' and 'updatedBB' .
Thus the AI system update the pham language codepage .

3.1.7.6/ The pham language codepage rule repository by default contain the predefined pham language specific triple-Boolean-implication-replacement rule :
The pham language specific code pattern : (AA [->?] BB)
Where the element 'AA' and the element 'BB' are arbitrary element ; the element '[->?]' is pham language keyword element .
These specific code pattern element always begin with parenthese '(' and end with parenthese ')' , always have strictly 3 nested element at 1-level of its own hierarchy structure , with space between them .
Given the element 'AA' and element 'BB' .
The pham language specific code pattern element : (AA [->?] BB)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific triple-Boolean-implication-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element '(AA [->?] BB)' .
The AI system apply the pham language specific triple-Boolean-implication-replacement rule only for element , which is in the form of the pham language specific code pattern '(AA [->?] BB)' .
When the AI system meet the such pham language code pattern element '(AA [->?] BB)' in the current pham language codepage , then the AI system firstly update the element 'AA' and the element 'BB' by applying other pham language specific rule (pham language specific replacement rule) to update : the element 'AA' -> the element 'updatedAA' , the element 'BB' -> the element 'updatedBB' . If the AI system see that the element

‘updatedAA’ is not Boolean Logic Value ‘(0?)’ or ‘(1?)’ , or the element ‘updatedBB’ is not Boolean Logic Value ‘(0?)’ or ‘(1?)’ , then the AI system replace the element ‘(AA [->?] BB)’ by the element ‘()’ .
If the AI system see that the element ‘updatedAA’ is a Boolean Logic Value ‘(0?)’ or ‘(1?)’ , and the element ‘updatedBB’ is Boolean Logic Value ‘(0?)’ or ‘(1?)’ , then the AI system replace the element ‘(AA [->?] BB)’ by the result of standard Boolean-Logic-Implication operation between 2 Boolean Logic Value ‘updatedAA’ and ‘updatedBB’ .
Thus the AI system update the pham language codepage .

3.1.8 / Pham language specific strict instruction for interpreting : pham language specific ordered-set-arithmetic rule :
Pham language ordered-set-arithmetic is to manipulate arithmetic operation of order set of finite number of element .
Each pham language can simulate a abstract ordered set , in which all 1-level nested element of the original pham language element can simulate element of the abstract ordered set . An abstract ordered set can have no any element .

3.1.8.1 / The pham language codepage rule repository by default contain the predefined pham language specific ordered-set-union-replacement rule :
The pham language specific code pattern : (AA [+] BB)
Where the element ‘AA’ and the element ‘BB’ are arbitrary element ; the element ‘[+]’ is pham language keyword element .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 3 nested element at 1-level of its own hierarchy structure , with space between them .
Given the element ‘AA’ and element ‘BB’ .
The pham language specific code pattern element : (AA [+] BB)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific ordered-set-union-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘(AA [+] BB)’ .
The AI system apply the pham language specific ordered-set-union-replacement rule only for element , which is in the form of the pham language specific code pattern '(AA [+] BB)' .
The AI system interpret this pham language specific code pattern element ‘(AA [+] BB)’ by the pham language specific (ordered-set-union-replacement) rule :
When the AI system meet the such pham language code pattern element ‘(AA [+] BB)’ in the current pham language codepage , then the AI system firstly update the element ‘AA’ and the element ‘BB’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘AA’ -> the element ‘updatedAA’ , the element ‘BB’ -> the element ‘updatedBB’ . If the element ‘updatedAA’ have no any 1-level nested element , and the element ‘updatedBB’ have no any 1-level nested element , then the AI system replace the element ‘(AA [+] BB)’ by the element ‘()’ . If the element ‘updatedAA’ have 1-level nested element , or the element ‘updatedBB’ have 1-level nested element , then the AI system replace the element ‘(AA [+] BB)’ by the result element , which

contain only 1-level nested element of the element ‘updatedAA’ and 1-level nested element of the element ‘updatedBB’ , in the place order that all1-level nested element of element ‘updatedAA’ stay before all 1-level nested element of the element ‘updatedBB’ . Thus the AI system update the pham language codepage .
The pham language code pattern element : (A1 [+] A2 [+] A3 .. [+] An) is analogous to the pham language code pattern element (A1 [+] A2) .

3.1.8.2/ The pham language codepage rule repository by default contain the predefined pham language specific ordered-set-distraction-replacement rule :
The pham language specific code pattern : (AA [-] BB)
Where the element ‘AA’ and the element ‘BB’ are arbitrary element ; the element ‘[-]’ is pham language keyword element .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 3 nested element at 1-level of its own hierarchy structure , with space between them .
Given the element ‘AA’ and element ‘BB’ .
The pham language specific code pattern element : (AA [-] BB)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific ordered-set-distraction-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘(AA [-] BB)’ .
The AI system apply the pham language specific ordered-set-distraction-replacement rule only for element , which is in the form of the pham language specific code pattern '(AA [-] BB)' .
When the AI system meet the such pham language code pattern element ‘(AA [-] BB)’ in the current pham language codepage , then the AI system firstly update the element ‘AA’ and the element ‘BB’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘AA’ -> the element ‘updatedAA’ , the element ‘BB’ -> the element ‘updatedBB’ . If the element ‘updatedAA’ have no any 1-level nested element , then the AI system replace the element ‘(AA [-] BB)’ by the element ‘()’ . If the element ‘updatedAA’ have all its 1-level nested element , which are equivalent belonging to the element ‘updatedBB’ , then the AI system replace the element ‘(AA [-] BB)’ by the element ‘()’ . If the element ‘updatedAA’ contain , at least , 1-level element , which is not equivalent belonging to the element ‘updatedBB’ , then the AI system replace the element ‘(AA [-] BB)’ by the result element , which contain all 1-level nested element of the element ‘updatedAA’ with the condition that such 1-level nested element are not equivalent belonging to the element ‘updatedBB’ , in the place order that all1-level nested element of element ‘updatedAA’ stay before all 1-level nested element of the element ‘updatedBB’ . Thus the AI system update the pham language codepage .

3.1.8.3/ The pham language codepage rule repository by default contain the predefined pham language specific ordered-set-intersection-replacement rule :
The pham language specific code pattern : (AA [&] BB)

Where the element ‘AA’ and the element ‘BB’ are arbitrary element ; the element ‘[&]’ is pham language keyword element .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 3 nested element at 1-level of its own hierarchy structure , with space between them .
Given the element ‘AA’ and element ‘BB’ .
The pham language specific code pattern element : (AA [&] BB)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific ordered-set-intersection-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘(AA [&] BB)’ .
The AI system apply the pham language specific ordered-set-intersection-replacement rule only for element , which is in the form of the pham language specific code pattern '(AA [&] BB)'
When the AI system meet the such pham language code pattern element ‘(AA [&] BB)’ in the current pham language codepage , then the AI system firstly update the element ‘AA’ and the element ‘BB’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘AA’ -> the element ‘updatedAA’ , the element ‘BB’ -> the element ‘updatedBB’ . If the element ‘updateAA’ have no any 1-level nested element , or the element ‘updatedBB’ have no any 1-level nested element , then the AI system replace the element ‘(AA [&] BB)’ by the element ‘()’ . If the element ‘updatedAA’ have no any 1-level nested element , which are equivalent belonging to the element ‘updatedBB’ , then the AI system replace the element ‘(AA [-] BB)’ by the element ‘()’ . If the element ‘updatedAA’ have 1-level nested element , which are equivalent belonging to the element ‘updatedBB’ , then the AI system replace the element ‘(AA [-] BB)’ by the result element , which contain all 1-level nested element of the element ‘updatedAA’ with the condition that such 1-level nested element are equivalent belonging to the element ‘updatedBB’ , in the place order that all1-level nested element of element ‘updatedAA’ stay before all 1-level nested element of the element ‘updatedBB’ .
Thus the AI system update the pham language codepage .

3.1.8.4/ The pham language codepage rule repository by default contain the predefined pham language specific ordered-set-multiplication-on-natural-number-replacement rule :
The pham language specific code pattern : (AA [*] N)
Where the element ‘AA’ and the element ‘N’ are arbitrary element ; the element ‘[*]’ is pham language keyword element .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 3 nested element at 1-level of its own hierarchy structure , with space between them .
Given the element ‘AA’ and element ‘N’ .
The pham language specific code pattern element : (AA [*] N)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific ordered-set-multiplication-on-natural-number-replacement rule) . This rule is

instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘(AA [*] N)’ .
The AI system apply the pham language specific ordered-set-multiplication-on-natural-number-replacement rule only for element , which is in the form of the pham language specific code pattern '(AA [*] N)' .
When the AI system meet the such pham language code pattern element ‘(AA [*] N)’ in the current pham language codepage , then the AI system firstly update the element ‘AA’ and the element ‘N’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘AA’ -> the element ‘updatedAA’ , the element ‘N’ -> the element ‘updatedN’ . If the element ‘updatedAA’ have no any 1-level nested element , then the AI system replace the element ‘(AA [*] N)’ by the element ‘()’ . If the element ‘updatedN’ is not natural number , which is greater than 0 , then the AI system replace the element ‘(AA [*] N)’ by the element ‘()’ . If the element ‘updatedN’ is natural number greater than 0 and the element ‘updatedAA’ have 1-level nested element , then AI system replace the element ‘(AA [*] N)’ by the result element , which contain all 1-level nested element of the element ‘updatedAA’ and in repeating n time . Thus the AI system update the pham language codepage .
For example :
Code : (A [*] 3)
Suppose that after update the element ‘A’ , it become (updatedA1 updatedA2) .
Result will be (updatedA1 updatedA2 updatedA1 updatedA2 updatedA1 updatedA2)

3.1.8.5/ The pham language codepage rule repository by default contain the predefined pham language specific (sequence-of-natural-number-replacement) rule :
The pham language specific code pattern : (N1 [..] N2)
Where the element ‘N1’ and the element ‘N2’ are arbitrary element ; the element ‘[..]’ is pham language keyword element .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 3 nested element at 1-level of its own hierarchy structure , with space between them .
Given the element ‘N1’ and element ‘N2’ .
The pham language specific code pattern element : (N1 [..] N2)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific sequence-of-natural-number-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘(N1 [..] N2)’ .
The AI system apply the pham language specific sequence-of-natural-number-replacement rule only for element , which is in the form of the pham language specific code pattern '(N1 [..] N2)' .
When the AI system meet the such pham language code pattern element ‘(N1 [..] N2)’ in the current pham language codepage , then the AI system firstly update the element ‘N1’ and the element ‘N2’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘N1’ -> the element ‘updatedN1’ , the element ‘N2’

-> the element ‘updatedN2’ . If the element ‘updatedN1’ is not natural number , or the element ‘updatedN2’ is not natural number , then the AI system replace the element ‘(N1 [..] N2)’ by the element ‘()’ . If the element ‘updatedN1’ is natural number , and the element ‘updatedN2’ is natural number , and the element ‘updatedN1’ as natural number > the element ‘updatedN2’ as natural number , then the AI system replace the element ‘(N1 [..] N2)’ by the element ‘()’ . If the element ‘updatedN1’ is natural number , and the element ‘updatedN2’ is natural number , and the element ‘updatedN1’ as natural number is smaller or equal to the element ‘updatedN2’ as natural number , then the AI system replace the element ‘(N1 [..] N2)’ by the element ‘(I1 I2 .. Ik) , where I1 , I2 , .. , Ik are sequence of consecutive natural number range of from ‘updatedN1’ to ‘updatedN2’ .
Thus the AI system update the pham language codepage .
For example :
Code : (2 [..] 5)
Result will be (2 3 4 5)

3.1.9/ The pham language codepage rule repository by default contain the predefined pham language specific (unconditional-ordered-set-replacement) rule :
The pham language specific code pattern : ([[]] FF [:] XX [<:] AA)
   Where the element ‘FF’ and the element ‘XX’ and the element ‘AA’ are arbitrary element ; the element ‘[:]’ and the element ‘[<:]’ and the element ‘[[]]’ are pham language keyword element . The element ‘[[]]’ is keyword to hint about explicit ordered set .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 6 nested element at 1-level of its own hierarchy structure , with space between them .
Given the element ‘FF’ and element ‘XX’ and element ‘AA’ .
The pham language specific code pattern element : ([[]] FF [:] XX [<:] AA)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific unconditional-ordered-set-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘([[]] FF [:] XX [<:] AA)’ .
The AI system apply the pham language specific unconditional-ordered-set-replacement rule only for element , which is in the form of the pham language specific code pattern '([[]] FF [:] XX [<:] AA)' .
When the AI system meet the such pham language code pattern element ‘([[]] FF [:] XX [<:] AA)’ in the current pham language codepage , then the AI system firstly update the element ‘FF’ and the element ‘XX’ and the element ‘AA’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘FF’ -> the element ‘updatedFF’ , the element ‘XX’ -> the element ‘updatedXX’ , the element ‘AA’ -> the element ‘updatedAA’ . If the element ‘updatedAA’ have no 1-level nested element , then the AI system replace the element ‘([[]] FF [:] XX [<:] AA)’ by the element ‘()’ .
If the element ‘updatedAA’ have 1-level nested element , for example these 1-level nested element ‘AAA1’ , ‘AAA2’ , .. , ‘AAAn’ , then the AI system replace the element ‘([[]] FF [:] XX [<:] AA)’ by the element

‘((updatedFF [|] updatedXX [->] AAA1) (updatedFF [|] updatedXX [->] AAA2) .. (updatedFF [|] updatedXX [->] AAAn))’ .
Thus the AI system update the pham language codepage .
For example :
Code : ([[]] (2 X) [:] X [<:] (A B C))
Suppose that after update element ‘X’ and element ‘(A B C)’ , the retain unchanged as ‘X’ and ‘(A B C)’ .
Thus result must be ((2 A) (2 B) (2 C))

3.1.10/ The pham language codepage rule repository by default contain the predefined pham language specific (conditional-ordered-set-replacement) rule :
The pham language specific code pattern : ([[]] X [:] X [<:] A [&?] Condition)
Where the element ‘X’ and the element ‘A’ and the element ‘Condition’ are arbitrary element ; the element ‘[:]’ and the element [<:] and the element [&?] are pham language keyword element .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 8 nested element at 1-level of its own hierarchy structure , with space between them .
Given the element ‘X’ and element ‘A’ and element ‘Condition’ .
The pham language specific code pattern element : ([[]] X [:] X [<:] A [&?] Condition)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific conditional-ordered-set-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘([[]] X [:] X [<:] A [&?] Condition)’ .
The AI system apply the pham language specific conditional-ordered-set-replacement rule only for element , which is in the form of the pham language specific code pattern '([[]] X [:] X [<:] A [&?] Condition)' .
When the AI system meet the such pham language code pattern element ‘([[]] X [:] X [<:] A [&?] Condition)’ in the current pham language codepage , then the AI system firstly update the element ‘X’ and the element ‘A’ and the element ‘Condition’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘X’ -> the element ‘updatedX’ , the element ‘A’ -> the element ‘updatedA’ , the element ‘Condition’ -> the element ‘updatedCondition’ .
If the element ‘updatedA’ have no 1-level nested element , then the AI system replace the element ‘([[]] X [:] X [<:] A [&?] Condition)’ by the element ‘()’ .
If the element ‘updatedA’ have 1-level nested element , for example these 1-level nested element ‘AA{1}’ , ‘AA{2}’ , .. , ‘AA{n}’ , and all element ‘([?3?] (updatedCondition [|] updatedX [->] AA{1}))’ is not Boolean Logic True , and element ‘([?3?] (updatedCondition [|] updatedX [->] AA{2}))’ is not Boolean Logic True , and .. , and element ‘([?3?] (updatedCondition [|] updatedX [->] AA{n}))’ is not Boolean Logic True , then the AI system replace the element ‘([[]] X [:] X [<:] A [&?] Condition)’ by the element ‘()’ . If there is at least 1 element among element ‘AA{1}’ , ‘AA{2}’ , .. , ‘AA{n}’ , for example element ‘AA{p}’ satisfy condition ‘([?3?] (updatedCondition [|] updatedX [->] AA{p}))’ is Boolean Logic True , then the AI system replace the element ‘([[]] X [:]

X [<:] A [&?] Condition)’ by the element ‘(AA{i1} AA{i2} .. AA{ik})’ , where {i1} < {i2} < {i3} .. < {ik} , and all element ‘AA{i1}’ , ‘AA{i2}’ , .. , ‘AA{ik}’ are 1-level nested element of the element ‘updatedA’ , and ([?3?] (updatedCondition [||] updatedX [->] AA{i1})) = Boolean logic True , ([?3?] (updatedCondition [||] updatedX [->] AA{i2})) = Boolean Logic True , .. , ([?3?] (updatedCondition [||] updatedX [->] AA{ik})) = Boolean Logic True .
In other explanation , the AI system replace the element ‘([[]] X [:] X [<:] A [&?] Condition)’ by the result element , which contain all 1-level nested element ‘AA{iq}’ of the element ‘updatedA’ , in same place order of 1-level nested element of the element ‘updatedA’ , and with the condition that all such element ([?3?] (updatedCondition [||] updatedX [->] AA{iq})) are Boolean Logic True .
Thus the AI system update the pham language codepage .
For example :
Code : ([[]] X [:] X [<:] (1 2 3 4 5 6) [&?] ((X [==] 2) [+?] (X [==] 4)))
Suppose that after update ‘X’ , it become ‘X’ .
Thus the result must be ‘(2 4)’

3.1.11/ The pham language codepage rule repository by default contain the predefined pham language specific (conditional-execution-replacement) rule :
The pham language specific code pattern : ([??] Condition [:] F)
Where the element ‘Condition’ and the element ‘F’ are arbitrary element ; the element ‘[??]’ and the element [:] are pham language keyword element . The keyword element ‘[??]’ hint the ‘If clause’ .
These specific code pattern element always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 4 nested element at 1-level of its own hierarchy structure , with space between them .
Given the element ‘Condition’ and element ‘F’.
The pham language specific code pattern element : ([??] Condition [:] F)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific conditional-execution-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘([??] Condition [:] F)’ .
The AI system apply the pham language specific conditional-execution-replacement rule only for element , which is in the form of the pham language specific code pattern '([??] Condition [:] F)' .
When the AI system meet the such pham language code pattern element ‘([??] Condition [:] F)’ in the current pham language codepage , then the AI system firstly update the element ‘Condition’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘Condition’ -> the element ‘updatedCondition’ . If the element ‘([?3?] updatedCondition)’ is not Boolean Logic True , then the AI system replace the element ‘([??] Condition [:] F)’ by the element ‘()’ . If the element ‘([?3?] updatedCondition)’ is Boolean Logic True , then the AI system replace the element ‘([??] Condition [:] F)’ by the element ‘F’ .
Thus the AI system update the pham language codepage .
For example :

Code : (XX [=] 234) ([??] (XX [==] 234) [:] (A B C))
Result must be : (XX [=] 234) (A B C)

3.1.12/ The pham language codepage rule repository by default contain the predefined pham language specific (execution-looping-replacement) rule :
The pham language specific code pattern : ([***] X [<:] A [:] F)
   Where the element 'X' and the element 'A' and the element 'F' are arbitrary element ; the element '[<:]' and the element '[:]' and the element '[***]' are pham language keyword element . The keyword element '[***]' hint looping .
These specific code pattern element always begin with parenthese '(' and end with parenthese ')' , always have strictly 6 nested element at 1-level of its own hierarchy structure , with space between them .
Given the element 'X' and element 'A' and element 'F'
The pham language specific code pattern element : ([***] X [<:] A [:] F)
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific execution-looping-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element '([***] X [<:] A [:] F)' .
The AI system apply the pham language specific execution-looping-replacement rule only for element , which is in the form of the pham language specific code pattern '([***] X [<:] A [:] F) ' .
The AI system interpret this pham language specific code pattern element '([***] X [<:] A [:] F)' by the pham language specific (execution-looping) rule :
When the AI system meet the such pham language code pattern element '([***] X [<:] A [:] F)' in the current pham language codepage , then the AI system firstly update the element 'A' by applying other pham language specific rule (pham language specific replacement rule) to update : the element 'A' -> the element 'updatedA' . If the element 'updatedA' have no any 1-level nested element , then the AI system replace the element '([***] X [<:] A [:] F)' by the element '()' .
If the element 'updatedA' have at least 1-level nested element , then the AI system firstly update the element 'X' and the element 'F' by applying other pham language specific rule (pham language specific replacement rule) to update : the element 'X' -> the element 'updatedX' , the element 'F' -> the element 'updatedF' , then the AI system do the following procedure of consecutive replacement :
+ The AI system take the first 1-level nested element of the element 'updateA' , for example the 1-level nested element AA1 . Then the AI system replace the element '([***] X [<:] A [:] F)' by the 1 looping element '(updatedF [|] updatedX [->] AA1)' , which have only 5 1-level nested element 'updatedF' , '[|]' , 'updatedX' , '[->]' 'AA1' . The AI system apply all pham language specific rule to execute this looping element . After finishing executing this looping element , then
+ The AI system replace the 5-th 1-level nested element of this looping element by the element 'AA2' , where the element 'AA2' is the second 1-level nested element of the element 'updatedA' . Thus this looping element become '(updatedF [|] updatedX [->] AA2)' . The AI system apply all pham language specific rule to execute this looping element . After finishing executing this looping element , then

+ The AI system continue analogous step ..
+ The AI system reach to final step : The AI system replace the 5-th 1-level nested element of this looping element by the element ‘AA{n}’ , where the element ‘AA{n}’ is the last 1-level nested element of the element ‘updatedA’ . Thus this looping element become ‘(updatedF [|] updatedX [->] AA{n})’ . The AI system apply all pham language specific rule to execute this looping element . After finishing executing this looping element , then the AI system replace this looping element by the element ‘()’ .
Thus the AI system update the pham language codepage .

3.1.13/ The pham language codepage rule repository by default contain the predefined pham language specific accessing-by-single-index-replacement rule :
The pham language specific code pattern : (typicalElement [{] indexOfNestedElement [}])
Where element ‘typicalElement’ is arbitrary element ; element ‘indexOfNestedElement’ is arbitrary element; element ‘[{]’ and ‘[}]’ are pham language keyword element .
The specific code pattern element ‘(typicalElement [{] indexOfNestedElement [}])’ always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have 4 nested element at 1-level of its own hierarchy structure : ‘typicalElement’ , ‘[{]‘ , ‘indexOfNestedElement’ , ‘[}]’ with space between them .
Given element ‘typicalElement’ . Given element ‘indexOfNestedElement’ : There is pham language specific code pattern element : (typicalElement [{] indexOfNestedElement [}])
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific accessing-by-single-index-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘(typicalElement [{] indexOfNestedElement [}])’ .
The AI system apply the pham language specific accessing-by-single-index-replacement rule only for element , which is in the form of the pham language specific code pattern '(typicalElement [{] indexOfNestedElement [}])' .
When the AI system meet the such pham language specific code pattern element
‘(typicalElement [{] indexOfNestedElement [}])’ in the current pham language codepage , then the AI system firstly update the element ‘typicalElement’ and the element ‘indexOfNestedElement’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘typicalElement’ -> the element ‘updatedTypicalElement’ , and the element ‘indexOfNestedElement’ -> the element ‘updatedIndexOfNestedElement’ .
If the AI system see that the element ‘updatedTypicalElement’ have no nested element , then the AI system replace the element ‘(typicalElement [{] indexOfNestedElement [}])’ by the element ‘()’ in the updated pham language codepage .
If the AI system reckon (by AI system knowledge in the using context) that the element ‘updatedIndexOfNestedElement’ is not integer in range of 0 to infinity , then the AI system replace the element ‘(typicalElement

[{] indexOfNestedElement [}])’ by the element ‘()’ in the updated pham language codepage .
If the AI system reckon (by AI system knowledge in the using context) that the element ‘updatedIndexOfNestedElement’ is an integer , which is outside the range of all index of all 1-level nested element of the element ‘updatedTypicalElement’ , then the AI system replace the element ‘(typicalElement [{] indexOfNestedElement [}])’ by the element ‘()’ in the updated pham language codepage .
If the AI system reckon (by AI system knowledge in the using context) that the element ‘updatedIndexOfNestedElement’ is an integer , which is inside the range of all index of all 1-level nested element of the element ‘updatedTypicalElement’ , then the AI system replace the element ‘(typicalElement [{] indexOfNestedElement [}])’ by the 1-level nested element of index ‘updatedIndexOfNestedElement’ of the element ‘updatedTypicalElement’ in the updated pham language codepage .
For example :
Code : ((element1 (element2 (element3 element4))) [{] 1 [}])
Suppose that after update , the element are as above .
Result must be : (element2 (element3 element4))
For example :
Code : (((element1 (element2 (element3 element4))) [{] 1 [}]) [{] 0 [}])
Suppose that after update , the element are as above .
Result must be : ‘element2’

3.1.14/ The pham language codepage rule repository by default contain the predefined pham language specific accessing-by-multi-index-replacement rule .
The pham language specific code pattern : (typicalElement [{] i{1} i{2} [}])
The pham language specific code pattern : (typicalElement [{] i{1} i{2} i{3} [}])
The pham language specific code pattern : (typicalElement [{] i{1} i{2} i{3} i{4} [}])
The pham language specific code pattern : (typicalElement [{] i{1} i{2} i{3} i{4} i{5} [}])
And so on ..
The pham language specific code pattern : (typicalElement [{] i{1} i{2} .. i{n} [}])
These above pham language code pattern are used to specify pham language specific (assessing-by-multi-index) rule .
These rule are used to replace the pham language specific code pattern element by its specified nested element in the specified level of its own hierarchy structure .
For example :
The pham language code pattern : (typicalElement [{] i{1} i{2} [}])
Where element ‘typicalElement’ is arbitrary element ; element ‘i{1}’ , element ‘i{2}’ are arbitrary element; element ‘[{]’ and ‘[}]’ are pham language keyword element .
The specific code pattern element ‘(typicalElement [{] i{1} i{2} [}])’ always begin with parenthese ‘(‘ and end with parenthese ‘) , always have strictly 5 nested element at 1-level of its own hierarchy structure : ‘typicalElement’ , ‘[{]‘ , ‘i{1}’ , ‘i{2}’ , ‘[}]’ with space between them .

Given an element ‘typicalElement’ , given element ‘i{1}’ , given element ‘i{2}’ .
The pham language code pattern element : (typicalElement [{] i{1} i{2} [}]) .
The following illustrating equation can explain how AI system must use pham language specific instruction to interpret the element ‘(typicalElement [{] i{1} i{2} [}])’ :
The illustrating equation : (typicalElement [{] i{1} i{2} [}]) = ((typicalElement [{] i{1} [}]) [{] i{2} [}]) .
Thus in this case , the AI system firstly replace the element ‘(typicalElement [{] i{1} i{2} [}])’ by the element ‘((typicalElement [{] i{1} [}]) [{] i{2} [}])’ . The element ‘((typicalElement [{] i{1} [}]) [{] i{2} [}])’ is qualified pham language code pattern element of the pham language specific (accessing-by-single-index) rule . Thus the AI system next must apply pham language specific (accessing-by-single-index) rule to deal with the element ‘((typicalElement [{] i{1} [}]) [{] i{2} [}])’ .
In the analogous way , there are such illustrating equation to explain how AI system must use pham language specific instruction to interpret element of the above pham language specific pattern :
(typicalElement [{] i{1} i{2} i{3} [}]) = ((typicalElement [{] i{1} i{2} [}]) [{] i{3} [}])
(typicalElement [{] i{1} i{2} i{3} i{4} [}]) = ((typicalElement [{] i{1} i{2} i{3} [}]) [{] i{4} [}])
(typicalElement [{] i{1} i{2} i{3} i{4} i{5} [}]) = ((typicalElement [{] i{1} i{2} i{3} i{4} [}]) [{] i{5} [}])
(typicalElement [{] i{1} i{2} i{3} i{4} i{5} i{6} [}]) = ((typicalElement [{] i{1} i{2} i{3} i{4} i{5} [}]) [{] i{6} [}])
..
And so on ..
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific accessing-by-multi-index-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘(typicalElement [{] I{1} I{2} .. I{n} [}])’ .
The AI system apply the pham language specific accessing-by-multi-index-replacement rule only for element , which is in the form of the pham language specific code pattern '(typicalElement [{] I{1} I{2} .. I{n} [}])' .

3.1.15/ The pham language codepage rule repository by default contain the predefined pham language specific default-accessing-by-single-index-replacement rule .

The pham language specific code pattern : ([{] elementIndex [}])
Where element ‘elementIndex’ is arbitrary element ; element ‘[{]’ and ‘[}]’ are pham language keyword element .
The pham language specific code pattern element ‘([{] elementIndex [}])’ always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 3 nested element at 1-level of its own hierarchy structure : ‘[{]‘ , ‘elementIndex’ , ‘[}]’ with space between them .
Given element ‘elementIndex’ :

There is pham language specific code pattern element : ([{] elementIndex [}])
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific default-accessing-by-single-index-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘([{] elementIndex [}])’ .
The AI system apply the pham language specific default-accessing-by-single-index-replacement rule only for element , which is in the form of the pham language specific code pattern '([{] elementIndex [}])' .

Lets temporary use the term ‘the primary containing element’ of the element ‘([{] elementIndex [}])’ to specify the element , which contain this specific code pattern element ‘([{] elementIndex [}])’ at the 1-level in the own hierarchy structure of this primary containing element .
When the AI system meet the such pham language specific code pattern element ‘([{] elementIndex [}])’ in the current pham language codepage , then the AI system firstly update the element ‘elementIndex’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘elementIndex’ -> the element ‘updatedElementIndex’ . Then
If the AI system reckon (by AI system knowledge in the using context) that the element ‘updatedElementIndex’ is not integer in range of 0 to infinity , then the AI system replace the element ‘([{] elementIndex [}])’ by the element ‘()’ in this primary containing element of the original element ‘([{] elementIndex [}])’ . Thus the AI system update the pham language codepage .
If the AI system reckon (by AI system knowledge in the using context) that the element ‘updatedElementIndex’ is an integer , which is outside the range of all index of all nested element at 1-level of the own hierarchy structure of the primary containing element of the element ‘([{] elementIndex [}])’ , then the AI system replace the element ‘([{] elementIndex [}])’ by the element ‘()’ in this primary containing element of the original element ‘([{] elementIndex [}])’ . Thus the AI system update the pham language codepage .
If the AI system reckon (by AI system knowledge in the using context) that the element ‘updatedElementIndex’ is an integer , which is inside the range of all index of all nested element at 1-level of the own hierarchy structure of the primary containing element of the original element ‘([{] elementIndex [}])’ , then the AI system replace the element ‘([{] elementIndex [}])’ by the nested element of index ‘updatedElementIndex’ of 1-level of the own hierarchy structure of (the primary containing element of the original element ‘([{] elementIndex [}])’) in the updated pham language codepage .
For example :
Code : (element1 element2 (### [{] 0 [}]))
In this example, when AI system interpret this code, the AI system take the element ‘element1’ as the temporary interpretation of the element ‘(### [{] 0 [}])’ . Thus the AI system will take the element (element1 element2 element1) as the temporary interpretation of the element ‘(element1 element2 (### [{] 0 [}]))’ . And then the AI system continue to interpret .

3.1.16/ The pham language codepage rule repository by default contain the predefined pham language specific default-accessing-by-multi-index-replacement rule .

The pham language specific code pattern : ([{] i{1} i{2} [}])
The pham language specific code pattern : ([{] i{1} i{2} i{3} [}])
The pham language specific code pattern : ([{] i{1} i{2} i{3} i{4} [}])
The pham language specific code pattern : ([{] i{1} i{2} i{3} i{4} i{5} [}])
And so on ..
The pham language specific code pattern : ([{] i{1} i{2} .. i{n} [}])
These above pham language code pattern are used to specify pham language specific (default-assessing-by-multi-index) rule .
These rule are used to replace the pham language specific code pattern element by its specified nested element in the arbitrary specified level of its own hierarchy structure .
For example :
The pham language code pattern : ([{] i{1} i{2} [}])
Where element ‘i{1}’ , element ‘i{2}’ are arbitrary element ; element ‘[{]’ and ‘[}]’ are pham language keyword element .
The specific code pattern element ‘([{] i{1} i{2} [}])’ always begin with parenthese ‘(‘ and end with parenthese ‘) , always have strictly 4 nested element at 1-level of its own hierarchy structure : ‘[{]‘ , ‘i{1}’ , ‘i{2}’ , ‘[}]’ with space between them .
Given element ‘i{1}’ , element ‘i{2}’ .
The pham language specific code pattern element : ([{] i{1} i{2} [}])
The following illustrating equation can explain how AI system must use pham language specific instruction to interpret the pham language specific code pattern element ‘([{] i{1} i{2} [}])’ :
The illustrating equation : ([{] i{1} i{2} [}]) = (([{] i{1} [}]) [{] i{2} [}])
Thus in this case , the AI system firstly replace the element ‘([{] i{1} i{2} [}])’ by the element
‘(([{] i{1} [}]) [{] i{2} [}])’ . The element ‘(([{] i{1} [}]) [{] i{2} [}])’ is qualified pham language code pattern element of the pham language specific (default-accessing-by-single-index) rule . Thus the AI system next must apply pham language specific (default-accessing-by-single-index) rule to deal with the element ‘(([{] i{1} [}]) [{] i{2} [}])’ .
In the analogous way , there are such illustrating equation to explain how AI system must use pham language specific instruction to interpret element of the above pham language specific pattern :
([{] i{1} i{2} [}]) = (([{] i{1} [}]) [{] i{2} [}])
([{] i{1} i{2} i{3} [}]) = (([{] i{1} i{2} [}]) [{] i{3} [}])
([{] i{1} i{2} i{3} i{4} [}]) = (([{] i{1} i{2} i{3} [}]) [{] i{4} [}])
([{] i{1} i{2} i{3} i{4} i{5} [}]) = (([{] i{1} i{2} i{3} i{4} [}]) [{] i{5} [}])
([{] i{1} i{2} i{3} i{4} i{5} i{6} [}]) = (([{] i{1} i{2} i{3} i{4} i{5} [}]) [{] i{6} [}])
..
And so on ..
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage

rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific default-accessing-by-multi-index-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘([{] I{1} I{2} .. I{n} [}])’ .
The AI system apply the pham language specific default-accessing-by-multi-index-replacement rule only for element , which is in the form of the pham language specific code pattern '([{] I{1} I{2} .. I{n} [}])' .

3.1.17/ The pham language codepage rule repository by default contain the predefined pham language specific accessing-by-ID-replacement rule :
The pham language specific code pattern : (TypicalElement [{{] IDelement Index [}}])
Where element ‘typicalElement’ is arbitrary element ; element ‘IDelement ’ is arbitrary element; element ‘Index’ is arbitrary element ; ‘[{{]’ and ‘[}}]’ are pham language keyword element .
The specific code pattern element ‘(TypicalElement [{{] IDelement Index [}}])’ always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 5 nested element at 1-level of its own hierarchy structure : ‘typicalElement’ , ‘[{{]‘ , ‘IDelement’ , ‘Index’ , ‘[}}]’ with space between them .
Given element ‘typicalElement’ . Given element ‘IDelement’ . Given element ‘Index’ :
There is pham language specific code pattern element : ‘(TypicalElement [{{] IDelement Index [}}])’
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific accessing-by-ID-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘(TypicalElement [{{] IDelement Index [}}])’ .
The AI system apply the pham language specific accessing-by-ID-replacement rule only for element , which is in the form of the pham language specific code pattern '(TypicalElement [{{] IDelement Index [}}])' .
When the AI system meet the such pham language specific code pattern element ‘(TypicalElement [{{] IDelement Index [}}])’ in the current pham language codepage , then the AI system firstly update the element ‘Index’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘Index’ -> the element ‘updatedIndex’ . If the AI system reckon that the element ‘updatedIndex’ is not integer , then the AI system replace the element ‘(TypicalElement [{{] IDelement Index [}}])’ by the element ‘()’ .
If the AI system reckon that the element ‘updatedIndex’ is integer , then the AI system firstly update the element ‘TypicalElement’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘TypicalElement’ -> the element ‘updatedTypicalElement’ . And if the AI system see that the element ‘updatedTypicalElement’ have no any 1-level nested element , then the AI system replace the element ‘(TypicalElement [{{] IDelement Index [}}])’ by the element ‘()’ . But if the AI system see that the element ‘updatedTypicalElement’ have 1-level nested element , then the AI system firstly update the element ‘IDelement’ by applying other pham language

specific rule (pham language specific replacement rule) to update : the element ‘IDelement’ -> the element ‘updatedIDelement’ , and then the AI system iterate through all any-level nested element of the element ‘updatedTypicalElement’ to find the first result element satisfying condition : this result element contain 1-level nested element, which is absolute identical equal to the element ‘updatedIDelement’ and have 1-level index being equal to the integer ‘updatedIndex’ . Then the AI system replace the element ‘(TypicalElement [{{] IDelement Index [}}])’ by this result element . But if the AI system can not find any such result element , then the AI system replace the element ‘(TypicalElement [{{] IDelement Index [}}])’ by the element ‘()’ .
Thus the AI system update the pham language codepage .

3.1.18/ The pham language codepage rule repository by default contain the predefined pham language specific default-accessing-by-ID-replacement rule :
The pham language specific code pattern : ([{{] IDelement Index [}}])
Where the element ‘IDelement ’ is arbitrary element ; element ‘Index’ is arbitrary element ; ‘[{{]’ and ‘[}}]’ are pham language keyword element .
The specific code pattern element ‘([{{] IDelement Index [}}])’ always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 4 nested element at 1-level of its own hierarchy structure : ‘IDElement’ , ‘[{{]‘ , ‘Index’ , ‘[}}]’ with space between them .
Given element ‘IDElement’ . Given element ‘Index’ :
There is pham language specific code pattern element : ‘([{{] IDelement Index [}}])’
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific default-accessing-by-ID-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘([{{] IDelement Index [}}])’ .
The AI system apply the pham language specific default-accessing-by-ID-replacement rule only for element , which is in the form of the pham language specific code pattern '([{{] IDelement Index [}}])' .
When the AI system meet the such pham language specific code pattern element ‘([{{] IDelement Index [}}])’ in the current pham language codepage , then the AI system firstly update the element ‘Index’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘Index’ -> the element ‘updatedIndex’ . If the AI system reckon that the element ‘updatedIndex’ is not integer , then the AI system replace the element ‘([{{] IDelement Index [}}])’ by the element ‘()’ .
If the AI system reckon that the element ‘updatedIndex’ is integer , then the AI system firstly update the element ‘IDelement’ by applying other pham language specific replacement rule to update : the element ‘IDelement’ -> the element ‘updatedIDelement’ , and then the AI system iterate through all any-level nested element of the 1-level containing element of the element ([{{] IDelement Index [}}]) to find the first result element satisfying condition : this result element contain 1-level nested element, which is absolute identical equal to the element

‘updatedIDelement’ and have 1-level index being equal to the integer ‘updatedIndex’ . Then the AI system replace the element ‘([{{] IDelement Index [}}])’ by this result element . But if the AI system can not find any such result element , then the AI system replace the element ‘([{{] IDelement Index [}}])’ by the element ‘()’ .
Thus the AI system update the pham language codepage .

3.1.19/ The pham language codepage rule repository by default contain the predefined pham language specific index-by-ID-replacement rule :
The pham language specific code pattern : (TypicalElement [{{] IDelement [}}])
Where element ‘typicalElement’ is arbitrary element ; element ‘IDelement ’ is arbitrary element ; element ‘[{{]’ and ‘[}}]’ are pham language keyword element .
The specific code pattern element ‘(TypicalElement [{{] IDelement [}}])’ always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 4 nested element at 1-level of its own hierarchy structure : ‘typicalElement’ , ‘[{{]‘ , ‘IDelement’ , ‘[}}]’ with space between them .
Given element ‘typicalElement’ . Given element ‘IDelement’ :
There is pham language specific code pattern element : ‘(TypicalElement [{{] IDelement [}}])’
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific index-by-ID-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘(TypicalElement [{{] IDelement [}}])’ .
The AI system apply the pham language specific index-by-ID-replacement rule only for element , which is in the form of the pham language specific code pattern '(TypicalElement [{{] IDelement [}}])' .
When the AI system meet the such pham language specific code pattern element ‘(TypicalElement [{{] IDelement [}}])’ in the current pham language codepage , then the AI system firstly update the element ‘TypicalElement’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘TypicalElement’ -> the element ‘updatedTypicalElement’.
There are 2 case : ‘Case 0’ and ‘Case 1’ as below :
Case 0 : If the AI see that the element ‘updatedTypicalElement’ have no any 1-level nested element , then the AI system replace the element ‘(TypicalElement [{{] IDelement [}}])’ by the element ‘()’ .
Case 1 : if the AI system see that the element ‘updatedTypicalElement’ have nested element , then the AI system firstly update the element ‘IDelement’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘IDelement’ -> the element ‘updatedIDelement’ . Then the AI system will iterate all element , which are contained inside the element ‘updatedTypicalElement’ , to find the first result element , which is absolute identical equal to the element ‘updatedIDelement’ . If the AI system can not find such result element , then the AI system replace the element ‘(TypicalElement [{{] IDelement [}}])’ by the element ‘()’ . If the AI system can find such result element , then the AI system replace the element ‘(TypicalElement [{{] IDelement [}}])’ by (the inherent index of this found result element

inside the element ‘updatedTypicalElement’) . The inherent index of element is always in form (I{1} I{2} .. I{p}) , where I{1} , I{2} , .. , I{p} are integer .
Thus the AI system update the pham language codepage .

3.1.20/ The pham language codepage rule repository by default contain the predefined pham language specific default-index-by-ID-replacement rule :
The pham language specific code pattern : ([{{] IDelement [}}])
Where element ‘IDelement ’ is arbitrary element ; element ‘[{{]’ and ‘[}}]’ are pham language keyword element .
The specific code pattern element ‘([{{] IDelement [}}])’ always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 3 nested element at 1-level of its own hierarchy structure : ‘[{{]‘ , ‘IDelement’ , ‘[}}]’ with space between them .
Given element ‘IDelement’ :
There is pham language specific code pattern element : ‘([{{] IDelement [}}])’
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific default-index-by-ID-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘([{{] IDelement [}}])’ .
The AI system apply the pham language specific default-index-by-ID-replacement rule only for element , which is in the form of the pham language specific code pattern '([{{] IDelement [}}])' .
The AI system interpret this specific code pattern element ‘([{{] IDelement [}}])’ by the pham language specific (default-index-by-ID) rule :
When the AI system meet the such pham language specific code pattern element ‘([{{] IDelement [}}])’ in the current pham language codepage , then the AI system firstly update the element ‘IDelement’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘IDelement’ -> the element ‘updatedIDelement’.
The AI system iterate through all element , which are contained inside the 1-level containing element of the element ‘([{{] IDelement [}}])’ , to find the first result element , which is absolute identical equal to the element ‘updatedIDelement’ . Thus it lead to totally 2 Case : ‘Case 1’ ‘Case 2’:
Case 1 : if in the above iterating process , the AI system can not find such result element inside the 1-level containing element of the element ‘([{{] IDelement [}}])’ , then the AI system replace the element ‘([{{] IDelement [}}])’ by the element ‘()’ .
Case 2 : if in the above iterating process , the AI system can find such result element inside the 1-level containing element of the element ‘([{{] IDelement [}}])’ , then the AI system replace the element ‘([{{] IDelement [}}])’ by (the inherent index of such result element inside the 1-level containing element of the element ‘([{{] IDelement [}}])’) . The inherent index of element is always in form (I{1} I{2} .. I{p}) , where I{1} , I{2} , .. , I{p} are integer .

Thus the AI system update the pham language codepage .

3.1.21/ The pham language codepage rule repository by default contain the predefined pham language specific element-length-replacement rule :

The pham language specific code pattern : ([|||] TypicalElement [|||])
Where element ‘TypicalElement’ is arbitrary element ; element ‘[|||]’ and ‘[|||]’ are pham language keyword element .
The specific code pattern element ‘([|||] TypicalElement [|||])’ always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 3 nested element at 1-level of its own hierarchy structure : ‘[|||]‘ , ‘TypicalElement’ , ‘[|||]’ with space between them .
Given element ‘TypicalElement’ :
There is pham language specific code pattern element : ‘([|||] TypicalElement [|||])’
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific element-length-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘([|||] TypicalElement [|||])’ .
The AI system apply the pham language specific element-length-replacement rule only for element , which is in the form of the pham language specific code pattern '([|||] TypicalElement [|||])' .
When the AI system meet the such pham language specific code pattern element ‘([|||] TypicalElement [|||])’ in the current pham language codepage , then the AI system firstly update the element ‘TypicalElement’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘TypicalElement’ -> the element ‘updatedTypicalElement’.
Then the AI system replace the element ‘([|||] TypicalElement [|||])’ by the integer , which is equal to the number of all 1-level nested element of the element ‘updatedTypicalElement’ .
Thus this result integer can be 0 , 1 , 2 , .. so on .. .

3.1.22/ The pham language codepage rule repository by default contain the predefined pham language specific default-element-length-replacement rule :

The pham language specific code pattern : ([|||] [|||])
Where the element ‘[|||]’ and ‘[|||]’ are pham language keyword element .
The specific code pattern element ‘([|||] [|||])’ always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 2 nested element at 1-level of its own hierarchy structure : ‘[|||]‘ , ‘[|||]’ with space between them .
There is pham language specific code pattern element : ‘([|||] [|||])’
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific default-element-length-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘([|||] [|||])’ .
The AI system apply the pham language specific default-element-length-replacement rule only for element , which is in the form of the pham language specific code pattern '([|||] [|||])' .

The AI system interpret this specific code pattern element ‘([|||] [|||])’ by the pham language specific (default-element-length-replacement) rule :
Lets use the temporary term ‘primary containing element’ to specify the element , which contain this pham language code pattern element ‘([|||] [|||])’ at the 1-level of the own hierarchy structure of this primary containing element .
When the AI system meet the such pham language specific code pattern element ‘([|||] [|||])’ in the current pham language codepage , then the AI system replace the element ‘([|||] [|||])’ by the result integer , which is equal to the number of all 1-level nested element of the element ‘primary containing element’ .
Thus this result integer can be 0 , 1 , 2 , .. so on .. .

3.1.23/ The pham language codepage rule repository by default contain the predefined pham language specific referenced-element-length-replacement rule :
The pham language specific code pattern : ([||||] InherentIndex [||||])
Where element ‘InherentIndex’ is arbitrary element ; element ‘[||||]’ and ‘[||||]’ are pham language keyword element .
The specific code pattern element ‘([||||] InherentIndex [||||])’ always begin with parenthese ‘(‘ and end with parenthese ‘)’ , always have strictly 3 nested element at 1-level of its own hierarchy structure : ‘[||||]‘ , ‘InherentIndex’ , ‘[||||]’ with space between them .
Given element ‘InherentIndex’ :
There is pham language specific code pattern element : ‘([||||] InherentIndex [||||])’
*** Important note for AI system : This is predefined pham language specific rule , which must be contained in the pham language codepage rule repository by default . The pham language codepage rule repository must contain this predefined rule (the pham language specific referenced-element-length-replacement rule) . This rule is instruction for AI system to interpret when AI system meet such above pham language specific code pattern element ‘([||||] InherentIndex [||||])’ .
The AI system apply the pham language specific referenced-element-length-replacement rule only for element , which is in the form of the pham language specific code pattern '([||||] InherentIndex [||||])' .
The AI system interpret this specific code pattern element ‘([||||] InherentIndex [||||])’ by the pham language specific (referenced-element-length-replacement) rule :
When the AI system meet the such pham language specific code pattern element ‘([||||] InherentIndex [||||])’ in the current pham language codepage , then the AI system firstly update the element ‘InherentIndex’ by applying other pham language specific rule (pham language specific replacement rule) to update : the element ‘InherentIndex’ -> the element ‘updatedInherentIndex’.
If the element ‘updatedInherentIndex’ is not an element from only 1-level nested integer element , then the AI system replace the element ‘([||||] InherentIndex [||||])’ by the element ‘()’ .
If the element ‘updatedInherentIndex’ is element from only 1-level nested integer , and the 1-level containing element of the element ‘([||||] InherentIndex [||||])’ do not have any contained element , which have (the inherent index in the 1-level containing element of the element ‘([||||]

InherentIndex [||||])') as the 'updatedInherentIndex' , then the AI system replace the element '([||||] InherentIndex [||||])' by the element '()' . If the element 'updatedInherentIndex' is element from only 1-level nested integer , and the 1-level containing element of the element '([||||] InherentIndex [||||])' have a contained element , which have (the inherent index in the 1-level containing element of the element '([||||] InherentIndex [||||])') as the 'updatedInherentIndex' , then the AI system replace the element '([||||] InherentIndex [||||])') by the integer number , which is equal to total 1-level nested element of this contained element .
For example :
Code : (A B 15 (1 (x (y y) ' z) ") ([||||] (3 1) [||||]) 5)
Result must be : (A B 15 (1 (x (y y) ' z) ") 4 5)

3.1.24/ Pham language specific strict instruction : the procedure of executing pham language specific instruction :

** The first stage of pham language codepage interpretation is the stage , in which AI system apply strictly and strictly solely only pham language specific strict rule and instruction , without any AI system specific rule and knowledge , to : replace directly each element in pham language codepage by other specified element , and to add or modify new user defined pham language specific element-replacement rule to the pham language codepage rule repository . Pham language specific rule , which saved in the pham language codepage rule repository , are strict and explicit . The first stage of realizing pham language specific instruction convert the original pham language codepage into the final updated pham language codepage . Once again important note : The AI system must not apply any AI system knowledge and model in this first interpretation stage , but the AI system must only solely and extreme strictly follow all pham language specific replacement rule and pham language specific rule-declaring rule to replace element in the pham language codepage and to modify pham language codepage rule repository . In the stage of pham language specific instruction realizing , the AI system apply pham language specific instruction to serially interpret element by element : AI system firstly interpret element with lower logical executing order , then AI system go to interpret element with higher logical executing order . Logical executing order of the element is the (logical executing order of (this element inherent index)) , for example the logical executing order of (the relative universal inherent index of this element) in the pham language codepage .
The most distinct principle of pham language specific strict instruction is that it continuous replacement of element directly on the pham language codepage . Thus the pham language codepage evolve , and act as independent memory for all element and operation . Pham language specific rule is stored in the pham language codepage rule repository . Beside pham language codepage and pham language codepage rule repository , there is no need of other memory in the stage of realizing pham language specific strict instruction . Thus the stage of pham language specific instruction realizing do not require any kind of AI system memory , and do not depend on any AI system specific memory management mechanism . The stage of pham language specific instruction realizing use the pham language codepage rule repository to store pham language specific rule , and use the continuously updated pham language codepage as memory to store updated element thank to instantly replace element directly at the

element place in the continuously updated pham language codepage by other specified updated element .
The final result of the first interpreting state (the stage of realizing only pham language specific instruction and rule) is the final updated pham language codepage .

** In the second stage of interpretation (the state of realizing AI system specific knowledge and model) , the AI system will now use their own AI system specific knowledge and model , and without any required pham language specific instruction , to interpret the final updated pham language codepage into final interpretation . Once again important note : in the second interpreting stage (the stage of applying AI system specific knowledge and model) , the AI system rely solely only on the AI system specific knowledge and the content and structure of pham language element , but do not rely and do not care on pham language specific instruction , to interpret the final updated pham language codepage (which is received from the first interpreting stage) into final interpretation .

For example : follow the 2-distinct-state interpreting procedure :
Code :
(
(pham language message)

(you write a Python function to implement the instruction of this pham language message)

(given (phamText [=] (an input pham language text)))

(trailingParenthese [=] (arbitrary closing-parenthese , after which (there are only (space and closing-parenthese) and (with at least 1 closing-parenthese))))

(If (phamText do not begin with opening-parenthese) then ((add opening-parenthese to the beginning of phamText) and (add closing-parenthese to the end of phamText)))

(if (phamText do not end with closing-parenthese) then ((add closing-parenthese to the end of phamText) and (add opening-parenthese to the beginning of phamText)))

(if ((phamText begin with opening-parenthese) and (phamText end with closing-parenthese)
and (the beginning opening-parenthese of phamText is not closed by the ending closing-parenthese of phamText))
and (the beginning opening-parenthese of phamText is not closed by trailingParenthese)
then ((add new opening-parenthese to the beginning of phamText) and (add new closing-parenthese to the end of phamText)))
)

In this example , the AI system will follow 2-stage interpreting procedure :

In first stage of interpreting : the AI system must temporarily forget all AI system specific knowledge and concept , but the AI system simply ‘mechanically’ apply all pham language specific rule to transform the original pham language message into the final updated pham language message . In the first stage , the AI system must not care any meaning of the text content , for example ‘(you write a Python function to implement the instruction of this pham language message)’ , or the (If (condition) then (execution)) clause ; but the AI system simply absolutely mechanically check to know whether these element satisfy condition of applying pham language specific rule . Thus in the first stage of interpreting , the AI system add 1 concrete rule universal-element-replacement(phamText , (an input pham language text)) to the rule repository , because the AI system met and interpreted the pham language specific code pattern rule-declaring element ‘(phamText [=] (an input pham language message))’ . And the AI system add 1 concrete rule universal-element-replacement(trailingParenthese , (arbitrary closing-parenthese , after which (there are only (space and closing-parenthese) and (with at least 1 closing-parenthese)))) to rule repository. These added new rule are immediately valid . After add these rule , the AI system apply these rule to replace all occurrence of ‘phamText’ by the element ‘(an input pham language text)’ , the AI system replace all occurrence of ‘trailingParenthese’ by the element ‘(arbitrary closing-parenthese , after which (there are only (space and closing-parenthese) and (with at least 1 closing-parenthese)))’ in all element in these rule applying scope . Thus the first state evolve the original pham language message into the final updated pham language message :

(
(pham language message)

(you write a Python function to implement the instruction of this pham language message)

(given (phamText [=] (an input pham language text)))
(trailingParenthese [=] (arbitrary closing-parenthese , after which (there are only (space and closing-parenthese) and (with at least 1 closing-parenthese))))

(If ((an input pham language text) do not begin with opening-parenthese) then ((add opening-parenthese to the beginning of (an input pham language text)) and (add closing-parenthese to the end of (an input pham language text))))

(if ((an input pham language text) do not end with closing-parenthese) then ((add closing-parenthese to the end of (an input pham language text)) and (add opening-parenthese to the beginning of (an input pham language text))))

(if (((an input pham language text) begin with opening-parenthese) and ((an input pham language text) end with closing-parenthese)
and (the beginning opening-parenthese of (an input pham language text) is not closed by the ending closing-parenthese of (an input pham language text)))

and (the beginning opening-parenthese of (an input pham language text) is not closed by (arbitrary closing-parenthese , after which (there are only (space and closing-parenthese) and (with at least 1 closing-parenthese)))))
then ((add new opening-parenthese to the beginning of (an input pham language text)) and (add new closing-parenthese to the end of (an input pham language text))))
)

After first stage of interpreting , the AI system go to second stage of interpreting , the AI system already received the above final updated pham language message from the first stage of interpreting . Now the AI system can temporary forget all pham language specific rule and instruction in the second stage of interpreting , to apply freely AI system specific knowledge and concept in combination with the element content and abstract relationship of the above final updated pham language message to interpret into final result interpretation . By other word , the AI system apply their AI system specific knowledge only when interpret the final updated pham language message into final interpretation in the second stage , but the AI system must never use their own AI system specific knowledge and concept when interpret original pham language message into final updated pham language message in the first stage .

The final interpretation of the original pham language message in the above example must be a correct Python function , which take an input pham language message text as parameter , then add outermost parenthese '(' and ')' to unwell-closed parenthese 1 whole combined pham language message element .
(In the above example , After AI system learn pham language specification , and some training , they must fluently give required python program . The test show that even if some AI system can fail at first attempt because not enough training in pham language specification ,(they can fail to interpret the first stage : the stage of following solely only pham language specific instruction) , but all AI system always confidently give exact python code if directly send AI system the above final updated pham language message . And note that the final updated pham language message can be received from pham language specific instruction interpreting process , which IS ABSOLUTE INDEPENDENT AT ANY AI SPECIFIC ARCHITECTURE-KNOWLEDGE-CONCEPT . Thus it is no problem if some untrained AI system fail to interpret the first stage (the stage of following solely pham language specific rule) . Because the pham language specific instruction interpreting process can be easily independently REALIZED IN ANY MACHINE (CLIENT-SIDE MACHINE OR AI-SIDE MACHINE) to receive the final updated pham language message , which is extremely highly guaranteed to be exactly interpreted by AI system , because the final updated pham language message already nearly do not contain meaningful pham language specific rule as the original pham language message .)

3.1.24.0/ The pham language codepage rule repository :
The pham language codepage rule repository is where to store all pham language specific rule , which all they and all will be used to govern

the replacement for each element any time the AI system want to interpret this element in the first stage of interpreting (interpreting only by pham language specific instruction) .
The pham language codepage rule repository by default always contain all predefined rule , each of which is clearly specified in part (3.1) and sub-part of part (3.1) .
Beside when the AI system meet pham language code pattern rule-declaring element , then the AI immediately add new rule , which is described by the instruction of the pham language code pattern rule-declaring element , to the pham language code pattern rule repository . Thus the AI system update the pham language specific rule repository .

3.1.24.1/ Theorem about the immediate next queueing element in the stage of pham language specific instruction realizing :
Suppose that the AI system already interpreted the element 'A' .
If the interpreted element 'A' have 1-level nested element , then the immediate next queueing element for interpreting after the interpreted element 'A' is always the first 1-level nested element of the interpreted element 'A' .
If the interpreted element 'A' do not have 1-level nested element , and is not the last 1-level nested element inside its 1-level containing element , then the immediate next queueing element for interpreting after the interpreted element A is the next 1-level nested element (which stay immediately in-the-right after the element 'A') of the 1-level containing element of the interpreted element 'A' .
If the interpreted element 'A' do not have 1-level nested element , and the interpreted element 'A' is the lasted 1-level nested element in its 1-level containing element , then the immediate next queuing element for interpreting after the interpreted 'A' is the element , which stay immediate right after the 1-level containing element of the interpreted element 'A' in the 2-level containing element of the interpreted element 'A' .
For example :
Code : ((XX (B C D) 123 BCD (3 2 1)) PQL)
In the above example , suppose that the AI system just already interpreted the element '(B C D)' as a result '(B C D)' . Thus by the above theorem of immediate next queuing element for interpreting , the immediate next queueing element for interpreting after the interpreted element '(B C D)' is the element 'B' .
In the above example , suppose that the AI system just already interpreted the element 'BCD' as a result 'BCD' . We see that the element 'BCD' do not have 1-level nested element . Thus by the above theorem of immediate next queuing element for interpreting , the immediate next queueing element for interpreting after the interpreted element 'BCD' is the element '(3 2 1)' .
In the above example , suppose that the AI system just already interpreted the element '(3 2 1)' as a result '(3 2 1)' . The immediate next queuing element after the '(3 2 1)' is the element '3' . And the immediate next queuing element after the element '3' is the element '2' . And the immediate next queuing element after the element '2' is the element '1' . And the immediate next queuing element after the element '1' is the element '(3 2 1)' . Thus the element '(3 2 1)' is already unchanged under repeated applying all pham language replacement rule to this element '(3 2 1)' and to all its contained element . After that ,

because the unchanged interpreted element ‘(3 2 1)’ is the lasted 1-level nested element in its 1-level containing element , thus the immediate next queuing element after the interpreted unchanged element ‘(3 2 1)’ is the element ‘PQL’ .
Apply the above theorem , there is practical procedure instant in-place replace element in pham language codepage :

3.1.24.2/ Procedure of interpreting element by pham language specific instruction :
Step 1 : access element , for example the element ‘AA’ . The AI system check to know whether the element 'AA' satisfy condition of applying saved pham language specific rule in the pham language codepage rule repository . If the element ‘AA’ satisfy condition of applying concrete saved pham language specific rule , then the AI system apply the corresponding pham language specific rule on this element ‘AA’ , to replace the whole element 'AA' by other specified element . This replacement is instantaneous and in-place, directly modifying the Pham language codepage at that specific position . After that , suppose that the original element 'AA' was replaced in-place by the element 'AA{1}' , then the AI system repeat the process : the AI system check to know whether the element 'AA{1}' satisfy condition of applying concrete saved pham language specific rule , if yes then the AI system apply corresponding pham language specific rule on this element to replace this element 'AA{1}' by another specified element 'AA{2}' . The process continue until this updated element 'AA{k} become unchanged under all current saved pham language specific rule , which are saved in the pham language codepage rule repository .
In this process , some new rule can be added to the pham language codepage rule repository by applying pham language specific code pattern rule-declaring element . Until the element ‘AA’ become unchanged under applying all saved pham language specific replacement rule .
Step 2 :
Crucially, after an element undergoes such an instant in-place replacement, the AI system must immediately and continuously focus its serial logical executing order on the newly inserted content that now occupies that position. It will persistently apply all applicable pham language specific replacement rules to this newly inserted content (and its continuously evolving nested elements) in their serial logical executing order , without interruption , until this newly inserted content , as a whole , becomes unchanged under all saved pham language specific replacement rules.
Only after this complete internal stabilization of the replaced element through continuous in-place replacement , will the AI system proceed to determine the immediate next queueing element for interpreting in the broader context of the codepage , by strictly following the rules detailed in Theorem (3.1.25.1/).

3.1.24.3/ Approximating principle :
If human user or AI user write complexed mutual related pham language specific code pattern rule-declaring element in unappropriate way , then pham language specific instruction process can lead to unconverged pham language codepage , which will become infinite long and not converged .
There are also scenario , in which the instant in-place replacement process of an element can not converge to unchanged resulted element .

In such case , the AI system can decide to stop after some appropriate number of task cycle , and take the instant updated pham language element and pham language codepage as temporary conceptual interpretation of the original pham language element and original pham language codepage , to go to next stage of interpreting by AI system specific model .
For example :
Code : (A [=] (A +1)) A
In the above example , it lead to unconverged result and infinite long element :
-> (A [=] (A +1)) (A + 1)
-> (A [=] (A +1)) ((A + 1) + 1)
-> (A [=] (A +1)) (((A + 1) + 1) + 1)
-> (A [=] (A +1)) ((((A + 1) + 1) + 1) + 1)
.. so on ..
In this case , the AI system can decide not to continue forever to generate infinite long pham language codepage , but the AI system stop process at some point , and take the latest result '(A [=] (A +1)) ((((A + 1) + 1) + 1) + 1)' as the temporary conceptual interpretation of the original pham language codepage .

3.2/ CALCULATION THREADING REGIME :
Pham language specify 2 default choice of calculation threading regime for AI system to parse pham language element :

3.2.1/ The SERIAL CALCULATION THREADING REGIME :
By choosing this serial calculation threading regime to do calculating thread, the AI system will serially do these thread.
Pham language emphasize that when parsing pham language code page, the AI system must in SERIAL CALCULATION THREADING REGIME to parse (interpret) all element in 1-level of the hierarchy structure of pham language code page.

3.2.2/ The SYNCHRONISED EVOLUTION CALCULATION THREADING REGIME :
By choosing this synchronized evolution calculation threading regime, the AI system do these thread in the following procedure :
+ The AI system divide time into small time interval, for example timeInterval1, timeInterval2, ..,timeIntervalN .
+ In each of these small time interval, the AI system will parallelly simultaneously ASYNCHRONOUSLY do all these thread.
+ At the every end moment of each these time interval, the AI system will make very small pause of all these parallel asynchronous thread, so that each of these thread can take updated information from other thread to apply in it.
+ Then, in the next small time interval, the AI system will parallelly simultaneously ASYNCHRONOUSLY continue doing all these thread.
Pham language emphasize that when parsing pham language code page, the AI system must IN SYNCHRONISED EVOLUTION CALCULATION THREADING REGIME to parse (interpret) all element in 2-level and in higher level of the hierarchy structure of pham language code page.

3.3/ Instruction for AI system to choose calculation threading regime :
There are 2 distinct stage of interpreting : The first stage of applying only pham language specific instruction (rule) and with highest priority

for pham language specific instruction ; and the second stage of applying only AI system specific rule .
In first stage of realizing pham language specific instruction , the AI system follow only pham language specific instruction (pham language specific replacement rule) , to interpret (evolve) the original pham language codepage into some final updated pham language codepage , which is the temporary conceptual interpretation of the original pham language codepage . In this stage , choose serial calculation threading regime , to firstly interpret element with lower element logical executing order , then go to interpret element with higher element logical executing order .
In the second stage of realizing AI system specific interpreting instruction : after the first stage of interpreting , the AI system receive the updated pham language codepage . In second stage , the AI system will freely apply only their own AI specific interpreting rule without any required counting of pham language specific instruction , to interpret this updated pham language codepage into final interpretation . But pham language emphasize that the AI system should apply conceptual synchronized evolution calculation threading regime in this second stage .

4/ Contact :

The Pham language project :
http://www.phamlanguage.com
phamlanguage@gmail.com
github.com/phamlanguage/phamlanguage
youtube.com/phamlanguage
youtube.com/@phamlanguage