Algonquin College Logo

# SCHOOL OF ADVANCED TECHNOLOGY

### ICT - Applications & Programming

### Computer Engineering Technology – Computing Science



A11

Language Specification

Team:

[Palwinder Kaur] - Id: [040996910] / [Hasibullah Yosufi ] - Id:[041012318]

Language Name [Hydra]

|  |  |
| --- | --- |
| **Part**  **1** | **Language User Reference** |

**EXPLANATION**

*Hydra is a python-format general-purpose programming language that has the* ***“.hd”*** *extension in its source file. .The Hydra language is easy to use on Windows, Mac, Linux with having the same grammatical and syntax format which further makes it more easier to learn and adapt*. *It is inspired by python, which is short, simple, and easy to read, however, hydra makes the simplicity of python, even more simpler and easier to read with a dynamically typed structure and only three simple data types (float, character, and bit). Hydra also enhances its execution efficiency, unlike python which uses an interpreter the execute line by line hydra takes the source code and executes it all in one go. This language takes the Boa source code and converts it to python programming language format which is more human-readable and user-friendly.*

* 1. **User Manual**

**Element 1: Name / Extension**

*The name of the language is* ***Hydra*** *which is named on an immortal, many-headed snake whose heads grew again as they were cut off, eventually killed by Hercules. Extension of the language is* ***.hd*** *in its source code. Hydra is a python-format general-purpose programming language and it has python like functionalities with more simpler way.*

**Element 2 – Comments**

*Hydra follows python-format for commenting but with a small difference on the source code. In order for the user to put a comment on something, they can simply start with putting the* ***/#*** *followed by the comment they want to put in the source code file and ends with* ***#/****. Same as python multi-line comments should all have to start with* ***#*** *followed by the user comment but as Hydra have minor difference in the representation* ***(/#).***

**Element 3 – Keywords**

*The lists of Hydra keywords are as follows:*

* ***print***  *(output the result into the console for the user)*
* ***input***  *(takes input from the keyboard)*
* ***or***  *(logical keyword or get true only if one of the condition is true)*
* ***and***  *(logical keyword and all the conditions should be true to become true)*
* ***not***  *(logical keyword not, inverse the condition)*
* ***if***  *(the if keyword is used in control statements to match )*
* ***elif*** *(else if keyword for the control statement)*
* ***else***  *(else control statement keyword if none of the conditions match)*
* ***for***  *(for control structure keyword to keep iterating till its range or its condition)*
* ***while*** *(while control structure keyword, keep iterating till the condition become false)*
* ***is***  *(to match if one variable or value is equal to another)*
* ***false*** *(logical data type false )*
* ***true*** *(logical data type true*
* ***break***  *(break get out a loop or any other control structure)*
* ***return***  *(return keyword end the function return the value of a function)*
* ***def***  *(def to define a function )*
* *Arithmetic operations, Calculations, etc, will be considered as usual with the other languages. ‘+’ for Addition, ‘-‘ for Subtraction, ‘/’ for division, ‘\*’ for multiplication.*

**Element 4 – Variables and Datatypes**

*Hydra has only three primitive data types. These data types are numeric, character, and bit (true/false).*

*The numeric data types are double precision which is four bytes and can be used for both integers and floating-point numbers. Character datatype can be both single characters or string of characters. Bit character holds only 0 or 1 value in it.*

**Element 5 – Variables and Datatypes**

*We can declare the variable in Hydra as shown below:*

* *For a numeric value we can just initialize the variable by using assignment operator.*

*Example:*

*a = 7.67 #/ “a” is a float variable and its value is 7.67.*

* *For a string value we can initialize the variable with assignment operator and put the string value in semicolon.*

*Example:*

*y = “Meghan” #/ “y” is a string variable ate its value is “Meghan”.*

***Element 6 - Commands***

* ***Attribution and assignment Statement****:*

*As we all know that when the arithmetic operation is going on, we assign the final value to a variable in order to print the output and the symbol for assigning it is “=” operator, variable should be on the left side and numeric value on right.*

*It is very important that associativity and order of precedence for any operation is +, - , \*, /, %.*

*Example:*

*/# declaring the variable/#*

*Num1 = 2*

*Num2 = 2.4*

*Sum = Num1 +Num2*

*Print(“ Total Sum is :” , Sum)*

* ***Selection****:*

*The Hydra supports the selection structure is the same as python using the keyword if, else, nested if-else. These conditional statements are used to run a specific statement among two or more. After the condition is set, we have to cover the block with colon only if the statements are more than 1 line if not then it is choice to use colon symbol.*

*Example:*

*/# variable initializing #/*

*x= 15*

*y= 12*

*/# If statement /#*

*If x == y:*

*print (“x is equal to y”)*

*/#elif statement/#*

*Elif x > y:*

*print(“x is greater than y”)*

*/# else statement #/*

*else:*

*print (“x is less than y”)*

***Interaction****:*

Iterative statements are the statements which are used when we need to repeat the loop again and again until the iteration condition does not turn false, this is also known as loops and the block covers in semicolon.

*Example:*

*/# while(Conditional statement): #/*

*/#loop will keep running again until condition comes false#/*

*i = 1*

*/# while loop for iterations #/*

*while i > 4 :*

*print (i)*

*i += 1*

*/# else statement /#*

*else:*

*print(“ i is no longer less* than 4” )

    
*For loop, the user can specify a range to keep iteration until that range*

*Example:*

* *For i in range(10):*

*print(i)*

* *for i in range(10,20)*

*print(i)*

* *for i in range (10,30,4)*

*print(i)*

***Input****:*

Every language needs to have a function which let the user to enter the information by the keyboard. Hydra have the keyword input(“”) and according to the data provided from keyboard, it will be stored in the variable with the value .

Example:

/# Taking the input from keyboard /#

value = input(“Enter your name:\n”)

print(‘Your name is {value}’)

***Output****:*

When the user is done working on the code, one can print the output on the screen with the help of using print(“”).

Example:

*/# print the output #/*

*print(“Hydra is a very excellent language”)*

***Functions****:*

To define a function, we use the def keyword followed by the function name and the return statement is dynamic. If the user does not include the return statement, the function will be void.

Example:

1.

/# define the function /#

def func\_name():

/#print the hello world message/#

Print (“Hello world”)

2.

/# define the function /#

def func\_name(a,b):

/# return the value /#

Return ( a+b)

**Element 7 – Proper elements**

*As Hydra is a Python like language, majority of its keyword are like Python such as print, input, Ture / False, and, or , not as others. But we are introducing several things in Hydra such as multi-line comments starting with /# and it ends with #/ sign. Furthermore, Hydra runs in one go as python run line by line, but Hydra run in one go. Most important feature which we introduce in Hydra is bit datatype which hold two value that is true/false or we can say that 0 and 1. Hydra’s extension is also unique that is .hd.*

|  |  |
| --- | --- |
| **Part**  **2** | **Examples** |

**Option 1: Python-like**

**Hello World**

|  |  |  |
| --- | --- | --- |
|  | */# Hello world example in Hydra language/#*  *print(“Hello World !!!!!”)* |  |

**Sphere Volume Expression**

|  |  |  |
| --- | --- | --- |
|  | /# Finding the volume of sphere#/  radius = float (input (‘ Enter the radius of sphere: ’)  volume = ( 4 \* 22 \* (radius \*\* 3)) / (3 \* 7 )  print (“ The volume of sphere is :” , round( volume, 2) ) |  |

|  |  |
| --- | --- |
| **Part**  **3** | **Architectural Aspects** |

**Advantages**

*Hydra is very fun and simple language. Hydra code’s compiler speed is faster as compared to interpreted languages because in those languages they need to have an interpreted to run the program on and they have to interpret every time they run, but in Hydra, because it is a compiler based language, have to compile for one time that’s it which eventually makes it faster.*

**Strategy: C Implementation**

*Hydra is a Python like language. For example if we wanted to implement Hydra into C , we will use C language data types and definitely plan high level of pseudocode. Furthermore, we can parse our language with defining the python like lexical resources. Definitely we are writing a complier for Hydra language.*

*In the language, we will define the variables and datatypes using def and struct in C language. Also, it will have the keyword input(“”) and according to the data provided from keyboard, it will be stored in the variable and this is how we can take input on the console. Also, with the help of initial lexical resources and tokens which we defined in our language, our complier will detect the input with the help of the functions and the For-loop, while loop conditional statements defined inside of the functions for each token and if the input do not match according to defined functions then it will generate error.*

**Basic ideas about C implementation**

*We are thinking to implement C- style data structures inside the language in which our more than one data variables will be declared and initialized inside that structure. The most hardest part while implementing hydra to C is parsing process and to be specific, the difficult part while implementing the parser will be writing the code of conditional statements inside the functions which plays the role of checking up the input with the lexical resources which are defined in the our language (Hydra) .*

**Problems when using C implementation**

The main problem which we will face during the implementation would be memory allocation and syntax .So for solving it we are using the malloc and calloc functions and if we talk about range of data types , all the data type which we will use in our language will already be in C language except one that is our specification of the language is bit data type which hold two values , True/False.

**Git Hub:**

***https://github.com/Khesorw/Compilers.git***

Algonquin College

Fall, 2022