(Dixit, 2023)

A person in a suit

Description automatically generated

ICL-1315 Introduction to Programming

TOBY BEEVERS

*Cambria ID: 21075206 Bangor ID: 500704883*

APPLIED DATA SCIENCE DEGREE APPRENTICESHIP BSC

Contents

[Abstract 2](#_Toc156675225)

[What are Variables? 2](#_Toc156675226)

[The use of variables in Java, Python, and C# 2](#_Toc156675227)

[Data Types 2](#_Toc156675228)

[Variable Naming 3](#_Toc156675229)

[Scope of Variables 3](#_Toc156675230)

[The use of conditional statements in Java, Python, and C# 4](#_Toc156675231)

[Syntax 4](#_Toc156675232)

[Statements 4](#_Toc156675233)

[The Growth Programming Languages 5](#_Toc156675234)

[References 6](#_Toc156675235)

[Appendix 7](#_Toc156675236)

[Program One 7](#_Toc156675237)

[Program One Testing 11](#_Toc156675238)

[Program Two 12](#_Toc156675239)

[Program Two Testing 15](#_Toc156675240)

# Abstract

The use of all three programming languages, Java, C# and Python is dependent on the use case, developers are encouraged to take time to understand the differences and apply these to the use case before jumping into development.

All three languages are Object Oriented with Python also being used for Machine Learning & AI, as well as automation and scripting of repetitive tasks. C# also offers Component Oriented programming which allows a slightly different approach to a large executable binary approach.

Some would say that Java is the only 100% Object Oriented Programming Languages, however it is argued by Anuradha Dixit on Coding Ninjas that Java’s backbone is not 100% Object Oriented, sighting the use of primitive data type like Boolean, Floats and Integers outside of Objects as the primary argument.

(Dixit, 2023)

# What are Variables?

Variables can be thought of as a container for storing things that will be manipulated or referred to by a computer program. A Variable is also named to help define the information held within it, the information must be standardised, and the variable can only store a single type of information.

## The use of variables in Java, Python, and C#

To begin with Python is defined as a Dynamic language whereas Java and C# are both considered Static Languages, the main difference being that Static Languages are required to be *Compiled* where they are converted to an executable, machine-readable, low-level code. Dynamic Languages are *Interpretated* and use an interpreter to convert them to machine readable code, this has the advantage of being able to use Python across operating systems.

### Data Types

Static Languages required their Variables to be *Declared* or explicitly defined at the point of creation and cannot be changed once defined. Dynamic Languages define their data types as they are being interpreted allowing a wider versatility particularly for beginners.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Integer** | **Boolean** | **String** |
| **Java** | int age = 25; | boolean isRaining = true; | String message = "Hello, world!"; |
| **C#** | int age = 25; | bool isRaining = true; | string message = "Hello, world!"; |
| **Python** | age = 25 | is\_raining = True | message = "Hello, world!" |

Within Python you can Cast your variable values enforcing the data types that are collected within them, this provides a similar approach to Java and C#, however Python is still interpreted in the same way.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Integer** | **Boolean** | **String** |
| **Python** | age = int(25) | is\_raining = bool(True) | message = string("Hello, world!") |

### Variable Naming

Naming conventions for variables is also an important consideration when looking across languages, whilst standard approaches of camelCase or PascalCase or snake\_case can be used, the different languages do have different requirements, some demand certain conventions others are less prescriptive but do have a best practice.

|  |  |  |
| --- | --- | --- |
|  | **Variables** | **Constraints** |
| **Java** | Uses camelCase  int myVariable; | Uses UPPERCASE  final int MAX\_VALUE = 100; |
| **C#** | Uses PascalCase  int MyVariable; | Uses PascalCase  const int MaxValue = 100; |
| **Python** | Uses snake\_case  my\_variable = 10 | Uses UPPERCASE  MAX\_VALUE = 100 |

*However, worth noting that due to my SQL background I use camelCase in Python and it works as well as snake\_case.*

All three languages follow the same principal restrictions for naming variables.

* Variable names cannot start with a digit like 9 or 3.
* Special characters other than the underscore are not permitted, example # % & etc.
* Keyword words like if, class, while, etc cannot be used as variable names as these are reserved.

(Oracle, 2023) (Python.Org, 2023) (Microsoft, 2023)

### Scope of Variables

Variable scopes determine where in a program a variable is accessible and usable, Java and C# use similar approaches whilst Python is totally different in its approach.

##### Java & C#

* **Local Variables:** Defined and accessible only within the declared scope.
* **Instance Variables:** Belong to an instance of a class and accessible throughout the class but unique to each instance.
* **Static Variables:** Belong to the class itself and shared among all instances of the class.

##### Python

* **Local Variables:** Defined within a function or method and accessible only within the function they are defined in.
* **Global Variables:** Defined outside functions or in the global scope and accessible throughout the module.

It could be argued that Static Variables in both C# and Java and Global Variables in Python are similar as they are both defined outside of the scope of the class itself and can be shared.

(Visual Studio Code, 2023)

# The use of conditional statements in Java, Python, and C#

Conditional Statements are a key part of writing a program. As the program follows the algorithm, conditional statements are used to make decisions for example if a value meets a certain criteria, then the answer is yes.

*Example*

a = 33  
b = 200  
if b > a:  
  print("b is greater than a")

The main difference between the three languages is their syntax, in terms of what they deliver in functionality it is exactly the same. It’s worth noting that the syntax differences do make them unique in their implementation and usage.

## Syntax

* Java uses curly braces {} to denote code blocks within conditional statements.
* C# also uses curly braces {} like Java for denoting code blocks within conditional statements.
* Python uses indentation to denote code blocks within conditional statements.

Unlike Java and C#, in Python indents are crucial for the structure and readability of the code.

Java and C# also require a semi colon to end each statement, this helps to prevent syntax ambiguities. Whereas Python simply allows a new line to end the statement.

## Statements

Both Java and C# allow for the use of a switch statement, this is lacking in Python often leading to long chains of if/elif to handle multiple conditions.

The switch statement in Java and C# allows the developer to define a variable containing the expression just once, the program than uses a case statement to validate the expression. Once the expression is validated as a match the block of code associated with the case statement will be executed.

In my humble opinion Python seems to be much easier language for beginners, the use of indenting increases Pythons readability allowing the reader to better understand the flow of the code. Python also use the *Interpreter* approach, which means as a beginner you do not need to understand the context of a complier allowing you to see your code results pretty much in real time which is fantastic for debugging.

# The Growth Programming Languages

A graph of a graph

Description automatically generated with medium confidence

(Tiobe.com, 2023)

Java has been a popular choice due to its “write once, run anywhere” approach, with the increase in mobile applications Java allows developers to re-use code across platforms. Java also has a rich community with plenty of open-source support, being owned by Oracle, Java also sees plenty of investment with regular releases every few years.

C# is a well-supported and distributed language by Microsoft within the .Net Framework. C# is a great choice for Web Application development, and thanks to the adoption of SOLID principles which are the 5 fundamentals of Object-Oriented Programming, C# is a popular chose with Developers.

(ByteHide.com, 2022)

Python however is a more all-round language, to begin with Python is much easier to understand for beginners and its ability to run without needing to understand compiling makes it perfect for learners. As you move into an intermediate use case, Python’s versatility to support everything from data analytics, sciences, to being able to build websites and even mobile technologies means that it’s becoming a go to for all users. Python also has a rich community, with access to many libraries and frameworks meaning that coders can get real support from community members who are able to share from experiences.

In my opinion Python is the most versatile coding language of the three, as it can be used to shape the data, as well as being able to run front and backend applications.

# References

ByteHide.com, 2022. *SOLID Principles in C#: A Complete Guide.* [Online]   
Available at: https://www.bytehide.com/blog/solid-principles-in-csharp  
[Accessed 17 01 2024].

Dixit, A., 2023. *Is Java an Object-Oriented language?.* [Online]   
Available at: https://www.codingninjas.com/studio/library/is-java-an-object-oriented-language  
[Accessed 28 12 2023].

Microsoft, 2023. *.NET documentation.* [Online]   
Available at: https://learn.microsoft.com/en-us/dotnet/  
[Accessed 28 12 2023].

Oracle, 2023. *Variables.* [Online]   
Available at: https://docs.oracle.com/javase/tutorial/java/nutsandbolts/variables.html#:~:text=A%20variable's%20name%20can%20be,%24%20%22%20or%20%22%20\_%20%22.  
[Accessed 28 12 2023].

Python.Org, 2023. *Type Variable Names.* [Online]   
Available at: https://peps.python.org/pep-0008/#naming-conventions  
[Accessed 28 12 2023].

Tiobe.com, 2023. *TIOBE Index for January 2024.* [Online]   
Available at: https://www.tiobe.com/tiobe-index/  
[Accessed 16 01 2023].

Visual Studio Code, 2023. *Programming Languages.* [Online]   
Available at: https://code.visualstudio.com/docs/languages/overview  
[Accessed 28 12 2023].

# Appendix

## Program One

A program that includes a selection of a variable type from a user, the use of IF statements and include at least one nested IF that will output the definition and usage of the selected variable type. **You are to research the variable definitions and paraphrase them into your own wording.**

*# /This is a simple Python program to collect a users name before presenting /#*

*# /info about Variables. The user can select through choices using Strings and Numbers /#*

*# /NB extra \*print ()\* statments have been used to add white space into the Terminal as there is no UI /#*

*# Ask the User their Name, collect this into a variable called name*

print () *#\**

name = input("What is your name? ")

print () *#\**

*# Now ask the user if they would like to know about Words, Numbers or something else*

print("Hello", name, "welcome to my program about Variables in Python.")

print("What would you like to learn about today?")

print() *#\**

*# Collect user response in a new variable called userChoice1 with some validation on the String*

userChoice1 = input("Type Words for Words, Numbers for Numbers, or Other for something else: ")

valid1 = '' *# This variable is empty, until the user enters a value.*

*# If the users entry matches the strings correctly*

*# it will set to True and end the While loop.*

while valid1 is not True:

  if userChoice1 == 'Words' or userChoice1 == 'Numbers' or userChoice1 == 'Other':

      valid1 = True

*# If the user enters the correct string then the variable will become True,*

*# if not it will loop through again*

      print() *#\**

  else:

      print () *#\**

      print (userChoice1, "is not a valid choice.")

      print ("By any chance did you miss a capital letter on your entry?")

      print () *#\**

      userChoice1 = input("Type Words for Words, Numbers for Numbers, or Other for something else: ")

*# These are the User choices, if the user selects "Other" then they will be asked to*

*# enter a new variable called userChoice2 which will enter the Nested If's*

*# This is the user selection for Words, it tells the user about using Strings in Python*

if userChoice1 == "Words":

      print ("You have selected to know more about Words, known as Strings or Text in Python!")

      print () *#\**

      print ("Strings in python are surrounded by either single quotation marks, or double quotation marks.")

      print ("'Hello' is the same as \"hello\" You can display a string literally with the print() function")

      print ("A str can be letters of numbers but it's important to understand that numbers")

      print ("stored in str can not be calculated!")

      print () *#\**

*# This is the user selection for Numbers, it tells the user about numbers in Python*

elif userChoice1 == "Numbers":

      print ("You have selected to know more about Numbers in Python!")

      print () *#\**

      print ("Numbers can be Integers, Floating Point Numbers, or Complex Numbers")

      print () *#\**

      print (" - Int, or integer, is a whole number, positive or negative, without decimal.")

      print (" - Float, or \"floating point number\" is a number, positive or negative,")

      print ("   containing one or more decimals.")

      print (" - Complex is an advanced number, which take a real number and an imaginary number.")

      print () *#\**

*# This the user slection for Other, this opens a Nest If with a While Loop.*

*# the user is asked to select further from 1,2,3,4 to find out more*

else:

      print ("You have selected to know more about something else in Python!")

      print () *#\**

      print ("Enter 1 to understand Sequence Types")

      print ("Enter 2 to understand Mapping Types")

      print ("Enter 3 to understand Set Types")

      print ("Enter 4 to understand Boolean Types")

      print () *#\**

*# Nested If with a While Loop*

      userChoice2 = input("Enter Your Selection between 1-4: ")

      valid2 = ''

*# While Loop is used to validate the variable userChoice2*

*# isdigit() is a built-in method for strings that returns True if all characters in the string are digits (numeric characters 0 to 9), or False*

      while valid2 is not True:

        if userChoice2.isdigit() and 1 <= int(userChoice2) <= 4:

            valid2 = True

        else:

            print () *#\**

            print(userChoice2, "is not a valid choice. Please try again.")

            print () *#\**

            userChoice2 = input("Enter Your Selection between 1-4: ")

*# These are the user choices if the user has selected Other > 1 or 2 or 3 or 4*

*# This Variable is Casting the data in the Variable userChoice2 to an Interger*

      userChoice2 = int(userChoice2)

*# This is the user selection for Sequence type*

      if userChoice2 == 1:

          print () *#\**

          print ("Thanks", name, "you selected", userChoice2)

          print ("So you want to know more about Sequence Types in Python!")

          print () *#\**

          print ("An advanced data type, the sequence data type in Python is used to store")

          print ("sequential data. The sequence data type in python can be considered")

          print ("as a container that can store different data.")

          print () *#\**

*# This is the user selection for Mapping type*

      elif  userChoice2 == 2:

          print () *#\**

          print ("Thanks", name, "you selected", userChoice2)

          print ("So you want to know more about Mapping Type in Python!")

          print () *#\**

          print ("An advanced data type, a mapping type is a data type comprised of a collection")

          print ("of keys and associated values. Python's only built-in mapping type is the")

          print ("dictionary. Dictionaries implement the associative array abstract data type.")

          print () *#\**

*# This is the user selection for Sets, List & Tuples*

      elif userChoice2 == 3:

          print () *#\**

          print ("Thanks", name, "you selected", userChoice2)

          print ("So you want to know more about Sets Types in Python!")

          print () *#\**

          print ("Sets, Tuples & Lists are all similar in that they are ordered collections of data")

          print (" - A Set is a collection of unique values stored in a single variable.")

          print ("   A Set is a collection which is unordered, unchangeable, and unindexed.")

          print () *#\**

          print (" - A Tuple is a collection of values that are ordered.")

          print ("   A Tuple is a collection which is ordered, unchangeable, and indexed.")

          print () *#\**

          print (" - A List is a collection of values that are ordered and change.")

          print ("   A List is a collection which is ordered, changeable, and indexed.")

          print () *#\**

*# This is the user selection for Boolean type*

      elif userChoice2 == 4:

          print () *#\**

          print ("Thanks", name, "you selected", userChoice2)

          print ("So you want to know more about Boolean Type in Python!")

          print () *#\**

          print ("In programming you often need to know if an expression is True or False,")

          print ("You can evaluate any expression in Python, and get either, True or False.")

          print ("When you compare two values, the expression is evaluated and")

          print ("Python returns the Boolean answer.")

          print () *#\**

*# The program will now ask the user if they would like to continue, this sets a Variable to Run or End the program.*

*# If the user enter a value that is not Yes or No (and see or statements), the While Loop will continue until the user enters the correct value.*

*# If the User enters YES the Variable is not used and the Program will use the exec and run again*

*# If the user enters NO the variable runEnd is set to True and will use exit*

runEnd = ''

while runEnd is not True:

  runEnd = input("Would you like to run this program again? ")

  if runEnd == "No" or runEnd == "no" or runEnd == "N" or runEnd == "n":

    runEnd = True

    print ("Thanks for using my Program")

    exit ()

  elif runEnd == "Yes" or runEnd == "yes" or runEnd == "Y" or runEnd == "y":

    print ("Lets Go again")

    exec(open("Assignment One - Program One (Variable Types with IF).py ").read())

*# The exec starment will need to contain the file name with .py as the extention*

  else:

    print () *#\**

    print(runEnd, "is not a valid choice. Please try again.")

    runEnd = '' *# This empty's the Variable to NULL so the While Loop will run again*

## Program One Testing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Test Type** | **Test Data** | **Reason** | **Expected Outcome** | **Actual Outcome** | **Pass/ Fail?** |
| 1 | Valid | Name = Dave Smith | Enter a valid name, this could be a string, number, or a null. | Valid | Valid | Pass |
| 2 | Valid | userChoice1 = Word | Enter Word and the program will continue to run and show info about Words | Valid | Valid | Pass |
| 4 | Invalid | userChoice1 = 100 | User will be told the selection is invalid and asked to try again | Invalid | Invalid | Pass |
| 5 | Valid | userChoice2 = 2 | User will be told about Mapping type | Valid | Valid | Pass |
| 5 | Invalid | userChoice2 = test | User will be told the selection is invalid and asked to try again. The number should be between 1 and 4 | Invalid | Invalid | Pass |
| 6 | Valid | runEnd = No | User will see Thanks for using my Program and the Program will close | Valid | Valid | Pass |
| 7 | Valid | runEnd = yes | User be asked to start again by entering their name | Valid | Valid | Pass |
| 8 | Invalid | runEnd = 100 | User will be told the selection is invalid and asked to try again | Invalid | Invalid | Pass |

## Program Two

A program that will carry out several forms of calculations based on user inputs and include IF statements that incorporate the ‘AND’ (or ‘OR’) operator.

*# /This is simple Python program that asks a user for the Date of Birth /#*

*# /The DoB is then used for various calculation to present back interesting facts /#*

*# /NB extra \*print ()\* statments have been used to add white space into the Terminal as there is no UI /#*

*# Begin by importing the Python DataTime Library, then set a variable with todays date*

import datetime

dateNow = datetime.datetime.now()

def handle\_invalid\_date\_input():

    print()

    print("Error: Invalid date format. Please enter your Date of Birth in the format DD-MM-YYYY.")

*# I did get some help on this error function from the Devs at work, however I did write*

*# this code myself they just guided me as I wanted something that I hadn't used before*

*# def is used to define a function, the funtion is then called further down the code*

*# https://www.w3schools.com/python/python\_functions.asp*

*# https://www.w3schools.com/python/python\_try\_except.asp#:~:text=The%20try%20block%20lets%20you,when%20there%20is%20no%20error.*

*# Now created another function for the main program*

def main():

*# Ask the User to enter their DOB, collect this into a Variable called userAge*

    print() *#\**

    print("Would you like to know some interesting facts about your age?")

    print() *#\**

    userAge = str(input("Enter your Date of Birth DD-MM-YYYY: "))

*# This function uses the Variable parseDate and collects the data from the Variable*

*# userAge before checking the date format is DD-MM-YYYY*

*# This code is attempting to parse a user's date in the format "DD-MM-YYYY" using the*

*# strptime method from the datetime library. The strptime method stands for "string parse time"*

*# and is used to convert a string representation of a date into a datetime object.*

    try:

        parsedDate = datetime.datetime.strptime(userAge, "%d-%m-%Y").date()

    except Exception as e:

        print(f"An unexpected error occurred: {e}")

        handle\_invalid\_date\_input()

        print() *#\**

        userAge = str(input("Enter your Date of Birth DD-MM-YYYY: "))

        return

*# Check for future dates*

    if parsedDate > dateNow.date():

        print("You entered a future date. Please enter a valid Date of Birth that is not in the future.")

        print() *#\**

        return

*# This variable stores the date after it has been converted into a usable date format*

*# using the datatime library > strptime function*

*# https://docs.python.org/3/library/datetime.html?highlight=datetime#module-datetime*

*# print (parsedDate) # Keep as debug test to see if the date format is parsed correctly*

*# Now start returning values*

    print() *#\**

    print("So you are: ")

*# Variable to store the age of the user in years*

    userYearsOld = dateNow.year - parsedDate.year *# Calculate the Users Age in Years*

    print("   \* " +str(userYearsOld) + " Years Old")

*# Variable to store the age of the user in days*

    numberOfDaysOld = (dateNow.date() - parsedDate).days *# Calculate the Users Age in Days*

    userDaysOld = numberOfDaysOld - dateNow.day *# Calculate the Users Age in Days*

    print("   \* That's " + str(userDaysOld) + " Days Old")

*# Add some calc to use if & and statements*

    if userYearsOld >= 18:

      print() *#\**

      print("And")

      print("   \* It looks like your old enough to vote!")

      print("   \* You can grab a beer at the bar!")

      print("   \* Plus you can also rent a car!")

    else:

      userAgeCalc = 18 - userYearsOld

      print() *#\**

      print("Not quite there, only " + str(userAgeCalc) + " years until you can vote & grab a beer!")

*#Add some trivia statements*

    print() *#\**

    print("Did you know:")

    print("   \* Your age in dog years is over " + str(userYearsOld \* 7))

    print("   \* Your age in cat years is over " + str(userYearsOld \* 4))

    print("   \* You probably had something like " + str(round(userDaysOld / 7)) + " Sunday Lunches!")

*#Add calcs about Driving ages if & and statements*

    print() *#\**

    print() *#\**

    print("What about Driving:")

    if userYearsOld < 16:

      print("    \* Sorry you can't start Driving just yet!")

      print() *#\**

    elif 16 <= userYearsOld <= 70:

      print("    \* Yep, you're good with driving as long as you've past your test!")

      print() *#\**

    else:

      print("    \* Did you know you need to be retesting every three years to keep you Licence?")

      print() *#\**

*# This is a While Loop to ask the user if they want to run the program again*

*# This checks if the Python script is being run as the main program and not imported as a module.*

*#When a Python script is executed, the interpreter assigns the special variable \_\_name\_\_ the value "\_\_main\_\_" if it is the main program.*

if \_\_name\_\_ == "\_\_main\_\_":

  while True:

      main()

      runEnd = input("Would you like to run this program again? ").lower()

      if runEnd not in {"yes", "y"}:

          print()

          print("Thanks for using my Program")

          break

## Program Two Testing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Test Type** | **Test Data** | **Reason** | **Expected Outcome** | **Actual Outcome** | **Pass/ Fail?** |
| 1 | Valid | userAge = 20-09-1975 | Enter a valid date format, the date will parse. Program will run. | Valid | Valid | Pass |
| 2 | Invalid | userAge = 09-20-1975 | Enter an invalid date format, still a date but not UK Standard | Invalid | Invalid | Pass |
| 4 | Invalid | userAge = 32-09-1975 | Enter a totally invalid date format | Invalid | Invalid | Pass |
| 5 | Invalid | userAge = Word | Enter a totally invalid format like a string | Invalid | Invalid | Pass |
| 5 | Valid | runEnd = y | Program will run again from the Main Function | Valid | Valid | Pass |
| 6 | Valid | runEnd = n | Program will close | Valid | Valid | Pass |