

# Programmimng in python, Jemma Williams 2206909

*by Jemma Williams*

---

**Submission date:** 12-Jan-2023 02:54PM (UTC+0000)

**Submission ID:** 194729869

**File name:**

41571\_Jemma\_Williams\_Programmimng\_in\_python\_Jemma\_Williams\_2206909\_466019\_465339096.odt  
(799.51K)

**Word count:** 2141

**Character count:** 12714

A variable in coding/programming is information (Value) that you wish to store in the program, (it is more beneficial also good practice to name your variable self-describing) a variable can be set by the programme creator or the user of the programme. In Python a variable name must start with a letter or an underscore and you cannot use special characters in the name of your variable. The name of a variable is also case sensitive, so if you were to have a variable named "holiday", you will need to use the exact same way you have typed it out, if you have only one variable named "holiday" and you were to type Holiday", "Holiday" is invalid as it does not exist in your programme. Or if you were to have two variables one being "holiday" and the second being "Holiday", if you are not precise in the way you type it, you may end up putting information some where in your programme that is not meant to be in that place.

In Python (and many other coding programmes) there are reserved words, you can not use a reserved word for a name of a variable.

A variable can be an integer, a float or a string. An integer variable is most likely to be used for whole numbers, a float is likely to be used for numbers that require decimals and a string is used for text. If you have a whole number as a variable but think later on in the programme it could be used in a "float" variable (which is common in calculations) you are able to "force" python to see the variable as a float by doing:

```
Number = float(12)
```

(number being the name of the variable, by typing float you are "casting" the variable as a float, and the number inside the bracket to say its this number we need this variable as)

When typing a variable name it is good practice to use "camel casing" this makes variables easier to read, so instead of naming a variable "myholiday" camel casing the name will be "myHoliday". Its helpful to see the words separated by an uppercase letter as spaces can not be used to name a variable. (using uppercase to "separate" the words from one another is known as camel casing). [1]

Python is very useful for, "if, elif (else if) and else". This will make the programme do something depending on the input and depending on the conditions set for the if, elif statement (else will not have a condition, else is only if the input does not satisfy the if or elif statements)

With these conditions, there does not always have to be an "elif", you can use just if and else, or you can have multiple "elif".

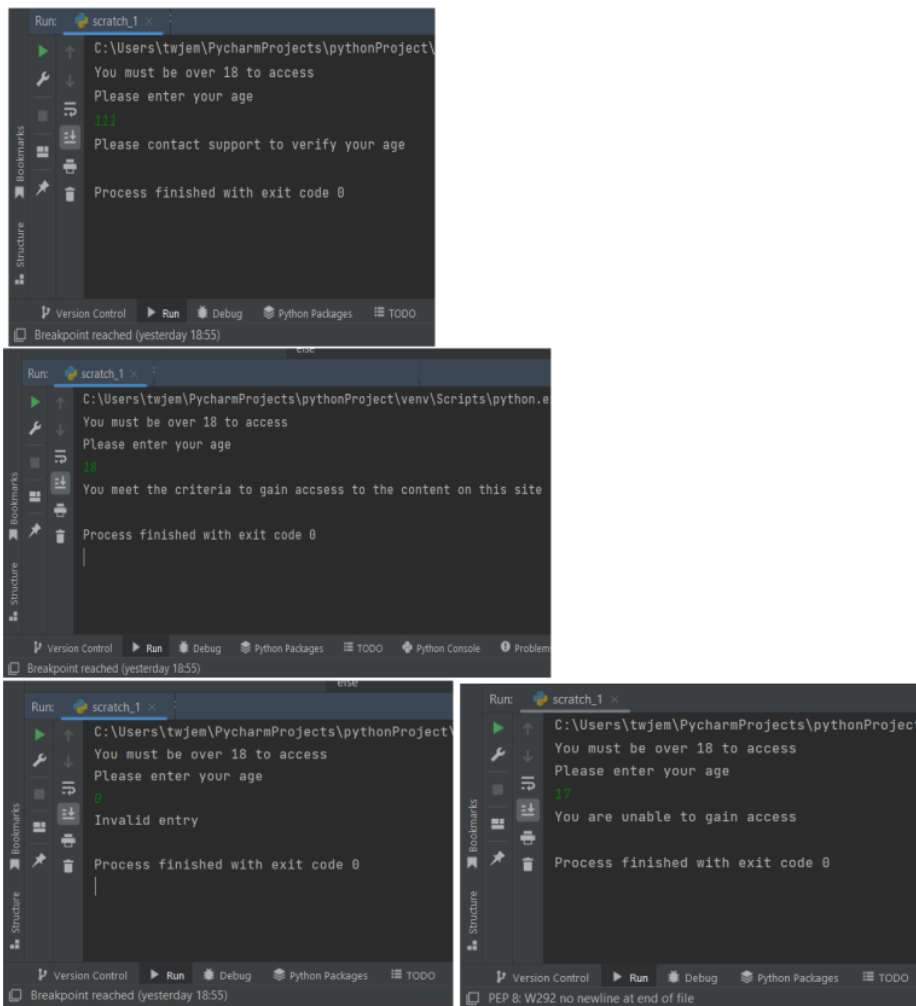
you can have more than one "if, elif and else" query, as long the correct if, elif and elses are indented to the same level and away from any other set of "if, elif and elses" [2]

Here are some examples of how, "if, elif and else" are helpful in a programme:

Example 1; Age restrictions.

```
print ("You must be over 18 to access")
print ("Please enter your age")
userAge = int(input())
if userAge >= 18 and userAge <= 110:
    print ("You meet the criteria to gain access to the content on this site")
elif userAge <= 17 and userAge >= 1:
    print ("You are unable to gain access")
elif userAge > 110:
    print ("Please contact support to verify your age")
else:
    print ("Invalid entry")
```

Here are the outcomes for the users different input:

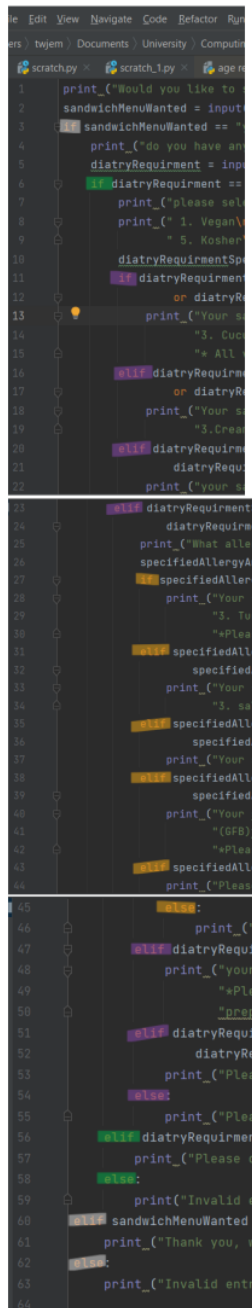


Example 2; Menu and dietary requirements.

```
print ("Would you like to see what sandwiches are available for you?\nPlease Answer Yes or No")
sandwichMenuWanted = input()
if sandwichMenuWanted == "yes" or sandwichMenuWanted == "Yes":
    print ("do you have any dietary requirments? \n Please Answer Yes or No")
    dietaryRequirment = input ()
    if dietaryRequirment == "yes" or dietaryRequirment == "Yes":
        print ("please select out of the following options: ")
        print (" 1. Vegan\n 2. Vegetairian\n 3. pescatarian\n 4. allergy/intolourence\n"
              " 5. Kosher\n 6. Other")
        dietaryRequirmentSpecified = input()
        if dietaryRequirmentSpecified == "1" or dietaryRequirmentSpecified == "Vegan" \
```

```
        or dietaryRequirmentSpecified == "vegan":
            print ("Your sandwich options are:\n1. Cucumber and lettuce\n2.
Tomatoe and lettuce\n"
                "3. Cucumber, spring onions, watercress and hoison
sauce\n"
                "** All vegan sandwiches are made with plant butter")
        elif dietaryRequirmentSpecified == "2" or dietaryRequirmentSpecified
== "vegetairian" \
            or dietaryRequirmentSpecified == "Vegetairian":
            print ("Your sandwich options are:\n1. Egg mayonaise\n2.Cheese
and watercress\n"
                "3.Cream cheese with herbs (parsley, orengano, rosemary
and thyme")
        elif dietaryRequirmentSpecified == "3" or dietaryRequirmentSpecified
== "Pescatarian" or \
            dietaryRequirmentSpecified == "pescatarian":
            print ("your sandwich options are:\n1. Tuna, mayonaise and
sweetcorn\n2. Salmon and cucumber\n3. cod")
        elif dietaryRequirmentSpecified == "4" or dietaryRequirmentSpecified
== "allergy/intolourence" or \
            dietaryRequirmentSpecified == "allergy" or
dietaryRequirmentSpecified == "intolourence":
            print ("What allergy/intolourence do you have?")
            specifiedAllergyAndIntolourence = input ()
            if specifiedAllergyAndIntolourence == "nut" or
specifiedAllergyAndIntolourence == "Nut" or specifiedAllergyAndIntolourence
== "nuts" or specifiedAllergyAndIntolourence == "Nuts":
                print ("Your sandwich choices are:\n1. Ham and
cheese\n2.Ham lettuce and cucumber\n"
                    "3. Turkey, lettuce and Bacon\n")
            elif specifiedAllergyAndIntolourence == "lactose" or
specifiedAllergyAndIntolourence == "lactose intolerance" or \
                specifiedAllergyAndIntolourence == "lactouse
intolerance":
                    print ("Your sandwich options are:\n1. Ham and lettuce\n2.
Turkey roll, bacon and lettuce\n3. salmon and cucumber\n *We only use plant
based butter for our lactose intolerance sandwiches*")
            elif specifiedAllergyAndIntolourence == "fish" or
specifiedAllergyAndIntolourence == "Fish" or \
                specifiedAllergyAndIntolourence == "Shellfish" or
specifiedAllergyAndIntolourence == "shellfish":
                    print ("Your sandwich options are:\n1. Ham and lettuce\n2.
Egg mayonaise\n3. Turkey, lettuce and bacon")
            elif specifiedAllergyAndIntolourence == "glueten intolerance"
or specifiedAllergyAndIntolourence == "Glueten intolerance" or \
                specifiedAllergyAndIntolourence == "Glueten
Intolourence" or specifiedAllergyAndIntolourence == "glueten":
                    print ("Your sanwich op;tions are:\n1.(GFB)Tuna, sweetcorn
and mayonaise\n2. "
                        "(GFB)turkey, bacon and lettuce\n3. (GFB)Ham and
cheese\n"
                        "**Please note all of our glutenfree sandwiches are
made with glueten free bread")
            elif specifiedAllergyAndIntolourence == "other" or
specifiedAllergyAndIntolourence == "option not given":
                    print ("Please contact us to see if we are able to
accomadate your needs, thank you.")
            else:
                print ("Please contact us")
        elif dietaryRequirmentSpecified == "5" or dietaryRequirmentSpecified
== "Kosher" or dietaryRequirmentSpecified == "kosher":
```

```
print ("your sandwich options are:\n1. (K)Beef\n2. (K)Turkey  
and lettuce\n3. (K)salmon and cucumber\n")  
    "*Please note all of our Kosher sandwiches and it's  
ingrediants are stored, "  
    "prepared and cooked in a different kitchen to our main  
kitchen, we also do not use any spread or butter for these sandwiches.")  
    elif dietaryRequirmentSpecified == "6" or dietaryRequirmentSpecified  
== "Other" or dietaryRequirmentSpecified == "other" or \  
        dietaryRequirmentSpecified == "requirement not listed" or  
dietaryRequirmentSpecified == "not listed":  
        print ("Please contact us to assist you further, thank you")  
    else:  
        print ("Please contact us for assistance")  
    elif dietaryRequirment == "No" or dietaryRequirment == "no":  
        print ("Please continue to see our full menu")  
    else:  
        print("Invalid entry,Pleasecontact us so we are able to assist  
you")  
elif sandwichMenuWanted == "No" or sandwichMenuWanted == "no":  
    print ("Thank you, we hope to see you soon.")  
else:  
    print ("Invalid entry, please restart")
```



```
1 print_("\nWould you like to\n2 sandwichMenuWanted = input\n3 sandwichMenuWanted == "\n4 print_("\ndo you have an\n5 dietaryRequirement = inp\n6 if dietaryRequirement ==\n7 print_("\nplease sel\n8 print_("\n1. Vegan\n9 " 5. Kosher\n10 dietaryRequirementSp\n11 if dietaryRequirement\n12 or dietaryRe\n13 print_("\nYour s\n14 "3. Cuc\n15 "4 All.\n16 elif dietaryRequire\n17 or dietaryRe\n18 print_("\nYour s\n19 "3. Creas\n20 elif dietaryRequire\n21 dietaryRequ\n22 print_("\nyour s\n23\n24 dietaryRequirement\n25 dietaryRequi\n26 print_("\nWhat all\n27 specifiedAllergyA\n28 specifiedAller\n29 print_("\nYour\n30 "3. Tur\n31 "+Plea\n32 elif specifiedAll\n33 specified\n34 print_("\nYour\n35 "3. sa\n36 elif specifiedAll\n37 specified\n38 print_("\nYour\n39 specified\n40 print_("\nYour\n41 "(GBB)\n42 "+Plea\n43 elif specifiedAll\n44 print_("\nPlease\n45\n46 else:\n47 print_("\n48 elif dietaryRequi\n49 print_("\nyour\n50 "+Plea\n51 "+prep\n52 elif dietaryRequi\n53 dietaryRe\n54 print_("\nPlease\n55\n56 else:\n57 print_("\nPlease c\n58\n59 else:\n60 print_("\nInvalid s\n61 sandwichMenuWanted\n62 print_("\nThank you, s\n63\n64 else:\n65 print_("\nInvalid entr
```

Here are screen shots of the right hand side of the code for a programme I have created (Menu & dietary requirements), here you can see there are a few "if", "elif" or "else" that belong to different variables. (I have highlighted them to the appropriate variable that they belong to).

Here there are four variables that require different outcomes depending on the user input, as explained earlier that for these "if", "elif" or/and "else" for each value it is required, the "set" that goes together need to be indented on the same line, I have used white to highlight the first set of "if", "elif" and "else" which is for the value I have named "sandwichMenuWanted", if the user answers yes, then the programme will ask the user has any dietary requirements, which then brings in the second set of "if", "elif" and "else" (highlighted in green) this value is named "dietaryRequirement", but if the user had input "no" into the value "dietaryRequirement" they would then receive a message saying: "please continue to see our full menu", the "else" part of this would be if the user had typed anything other than what the value equals to, (if the value equals yes it gives message "A" if the value equals no, it gives message "B" but if the value has been set by the user with anything else even if the user has set the value to nothing, no numbers, no characters ect, even if it is a spelling mistake or the user hasn't used a capital letter in their answer (If the programmer has not specified it will accept the answer with or without a capital letter) it will give message "C", which in this case the message for the value equalling to anything else other than what is stated, the message will read "Invalid entry, please restart"

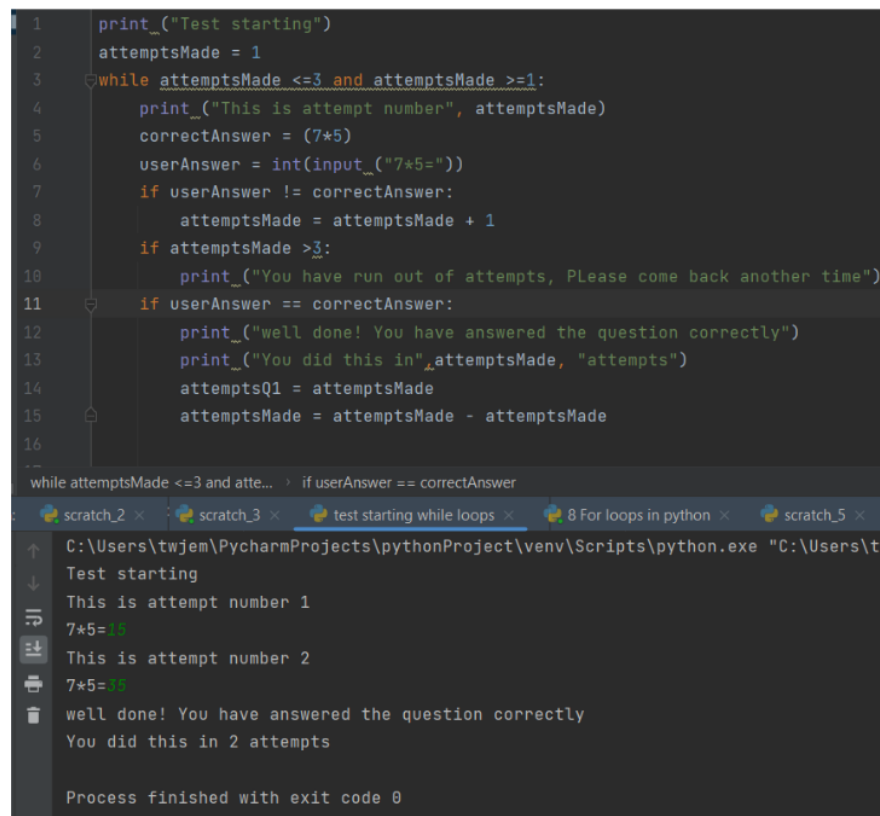
You will also notice that in my screenshots although my variable names have spelling mistakes such as, dietaryRequirement, as long as the programmer sticks to the same name/spelling as when first writing the variable python will understand what the value is of said variable (though it is probably better to spell it correctly so there is no confusion later on for themselves or for other programmers. It is also beneficial to use correct spelling for programming especially in a programme like one to take stock of medication, when names of medication are so alike avoid confusion or mistakes. [2]

In python you can use loops. Loops are useful for when an answer/input or "value" does not meet the requirement of the variable. While a variable that a user can input value, is not the value required (or incorrect value) the programme will be able to reset so the variable will have no value, and the user is able to input another value in the variable without resetting or restarting. This function is useful in on screen tests, so that the user will be unable to go onto further questions until

the value of the variable is equal to the value that python needs it to be to go further on in the programme. [3]

One example of a while loop is:

```
print ("Test starting")
attemptsMade = 1
while attemptsMade <=3 and attemptsMade >=1:
    print ("This is attempt number", attemptsMade)
    correctAnswer = (7*5)
    userAnswer = int(input ("7*5="))
    if userAnswer != correctAnswer:
        attemptsMade = attemptsMade + 1
    if attemptsMade >3:
        print ("You have run out of attempts, Please come back another
time")
    if userAnswer == correctAnswer:
        print ("well done! You have answered the question correctly")
        print ("You did this in",attemptsMade, "attempts")
        attemptsQ1 = attemptsMade
        attemptsMade = attemptsMade - attemptsMade
```



```
1 print_("Test starting")
2 attemptsMade = 1
3 while attemptsMade <=3 and attemptsMade >=1:
4     print_("This is attempt number", attemptsMade)
5     correctAnswer = (7*5)
6     userAnswer = int(input_("7*5="))
7     if userAnswer != correctAnswer:
8         attemptsMade = attemptsMade + 1
9     if attemptsMade >3:
10        print_("You have run out of attempts, Please come back another time")
11    if userAnswer == correctAnswer:
12        print_("well done! You have answered the question correctly")
13        print_("You did this in",attemptsMade, "attempts")
14        attemptsQ1 = attemptsMade
15        attemptsMade = attemptsMade - attemptsMade
16
```

while attemptsMade <=3 and atte... if userAnswer == correctAnswer

scratch\_2 x scratch\_3 x test starting while loops x 8 For loops in python x scratch\_5 x

C:\Users\twjem\PycharmProjects\pythonProject\venv\Scripts\python.exe "C:\Users\tw...

Test starting

This is attempt number 1

7\*5=15

This is attempt number 2

7\*5=10

well done! You have answered the question correctly

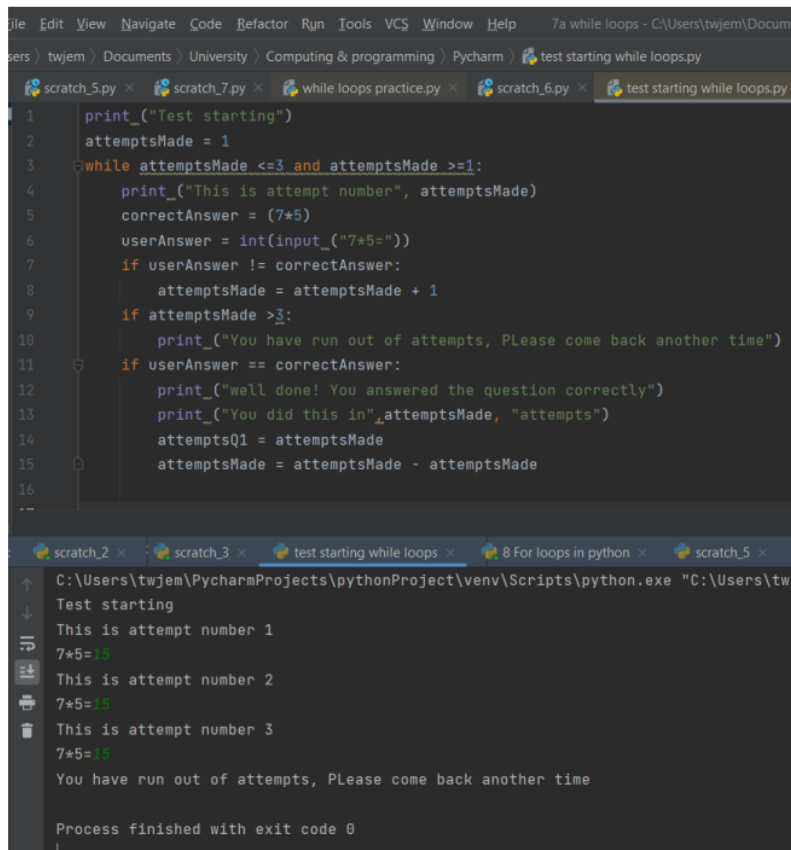
You did this in 2 attempts

Process finished with exit code 0

In my example the page before you will see, I have made a variable and names it "attemptsMade" the reason I have had it to the value of 1 is because I have done my example as an onscreen test, that you are only able to attempt so many times; (I have stopped the retry after a third attempt is made.).

The loop is able to stop if other statements are true. If the variable attemptsMade reaches over 3 the loop will stop and print a message. [4]

Though it was not a guarantee that the user will use all three attempts and not answer correctly, that is why I have a statement for if the question gets answered correctly. Within this statement I have requested python to print the amount of attempts were made, I have also created another variable so the information for "attemptsMade" will be stored in a different variable, but you will notice at the bottom, it will say attemptsMade = attemptsMade – attemptsMade so that attemptsMade will have the value of 0. The reason I have done this is because as you can see in the example on the previous page, the user has only used up two attempts, which is true against the "while" statement. Resetting the attemptsMade to 0 then makes it a false statement against the statement.



The screenshot displays the PyCharm IDE with a Python script and its execution output. The script, located in 'test starting while loops.py', uses a while loop to prompt the user for the answer to 7\*5. It tracks the number of attempts (attemptsMade) and includes an if statement to handle the case where the user runs out of attempts. The output window shows the program's execution, including the prompts, user input, and the final message after three failed attempts.

```
1 print_("Test starting")
2 attemptsMade = 1
3 while attemptsMade <=3 and attemptsMade >=1:
4     print_("This is attempt number", attemptsMade)
5     correctAnswer = (7*5)
6     userAnswer = int(input_("7*5="))
7     if userAnswer != correctAnswer:
8         attemptsMade = attemptsMade + 1
9     if attemptsMade >3:
10        print_("You have run out of attempts, PLease come back another time")
11    if userAnswer == correctAnswer:
12        print_("well done! You answered the question correctly")
13        print_("You did this in", attemptsMade, "attempts")
14        attemptsQ1 = attemptsMade
15        attemptsMade = attemptsMade - attemptsMade
16
```

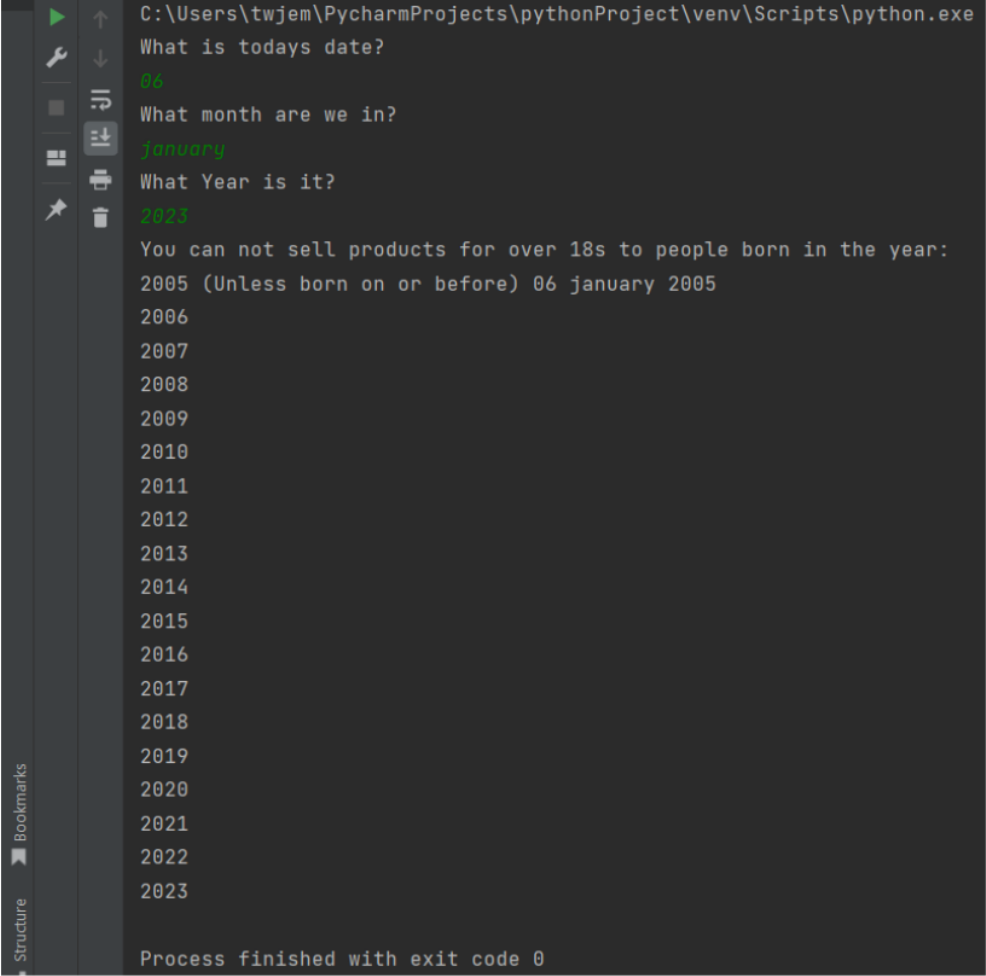
Output:

```
C:\Users\twjem\PycharmProjects\pythonProject\venv\Scripts\python.exe "C:\Users\twjem\PycharmProjects\pythonProject\venv\Scripts\python.exe"
Test starting
This is attempt number 1
7*5=13
This is attempt number 2
7*5=13
This is attempt number 3
7*5=13
You have run out of attempts, PLease come back another time
Process finished with exit code 0
```

Here I have added an If, the reason I have used an If statement rather than a "for" statement is because it was not a guarantee the user will use all three attempts. statement to go with my While statement so that when attemptsMade goes over the count of 3, It will have an end statement.

The next example will have a "For" loop. For loops tend to be used for statements that you know how many loops are needed.

```
print ("What is todays date?")
todaysDate = input()
print ("What month are we in?")
monthWeAreIn = input()
print ("What Year is it?")
thisYear = int(input())
print ("You can not sell products for over 18s to people born in the
year:")
print(thisYear - 18, "(Unless born on or before)", todaysDate,
monthWeAreIn, thisYear-18)
for thisYear in range (thisYear - 17, thisYear+1):
    print (thisYear)
```



The screenshot shows the PyCharm IDE interface. The top toolbar contains icons for running, debugging, and other development tools. The main editor window displays the output of a Python script. The script prompts the user for a date, month, and year. The user has entered '06', 'january', and '2023' respectively. The script then prints a message: 'You can not sell products for over 18s to people born in the year: 2005 (Unless born on or before) 06 january 2005'. Below this message, a list of years from 2005 to 2023 is printed. The bottom status bar indicates 'Process finished with exit code 0'.

```
C:\Users\twjem\PycharmProjects\pythonProject\venv\Scripts\python.exe
What is todays date?
06
What month are we in?
january
What Year is it?
2023
You can not sell products for over 18s to people born in the year:
2005 (Unless born on or before) 06 january 2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
Process finished with exit code 0
```

#### Exercise 1.

```
print ("Welcome to your family budgeting tool")
print ("are you logging wages or pocket money?")
systemSelection = input()
```

```
if systemSelection == ("wages" or systemSelection == "Wages"):
    print ("Please enter your full name")
    name = input()
    print("hello", name, "Please enter your age")
    age = int(input())
    if age<=15:
        print ("You are to young to use this system, please restart and
select the 'pocket money' link")
    elif age >= 16:
        print ("Please enter your National insurance number", name, ".")
        niNumber = input()
        payMistake = "unconfirmed"
        while payMistake == "yes" or payMistake == "YES" or payMistake ==
"Yes" or payMistake == "unconfirmed":
            print ("How much pay did you receive?")
            payReceived = float(input())
            if payReceived >=6000:
                print("That is higher than your usual income, is this a
mistake? (Please type Yes or No)")
                payMistake = input()
            if payReceived <= 5999 or payMistake == "no" or payMistake ==
"NO" or payMistake == "No":
                print ("please confirm if all details are correct:\nYou
have selected:",systemSelection,"\nYour name:",name,"\nYour age:",
age,"\nYour national Incurance number:", niNumber)
                print ("Please type yes or no")
                detailsCorrect = input()
                if detailsCorrect == "yes" or detailsCorrect == "Yes" or
detailsCorrect == "YES":
                    payMistake = "confirmed"
                    payReceived = "logged"
                    print ("Your wages have been stored in the family budget
system. thankyou!")
                elif detailsCorrect == "no" or detailsCorrect == "NO" or
detailsCorrect == "No":
                    print ("restarting family budget system")
```

#### Exercise 2.

```
squareExceedsCircle = "Yes"
pi = 3.14159265
while squareExceedsCircle == "Yes":
    squareExceedsCircle ="Waiting for input"
    print("Please enter the radius of circle in cm")
    circleRadius = int(input())
    circleArea = pi*circleRadius*circleRadius
    print("Please enter the width of the square in cm")
    squareWidth = int(input())
    squareArea = squareWidth*squareWidth
    if squareArea > circleArea:
        squareExceedsCircle = "Yes"
        print ("Please check your measurements and reenter")
    else:
        answer = round(circleArea-squareArea, 2)
        print ("The area of your shape is", answer,"cm")
```

[5]

#### Exercise 3.

```
import turtle
aT = turtle.Turtle()
aT.forward(100)
```

```
aT.right(120)
aT.forward(130)
aT.left(120)
aT.forward(100)
aT.right(90)
aT.forward(15)
aT.right(90)
aT.forward(125)
aT.right(120)
aT.forward(130)
aT.left(120)
aT.forward(75)
aT.right(87)
aT.forward(15)

aT.penup()
aT.goto(300,-50)
aT.pendown()
aT.circle(50)
aT.penup()
aT.goto(400,-50)
aT.pendown()
aT.circle(50)
aT.penup()
aT.goto(280,-50)
aT.pendown()
aT.circle(30)
aT.penup()
aT.goto(380,-50)
aT.pendown()
aT.circle(30)
```

[6]

## References

- [1] G. Thomas, "UWTSD Moodle, Python Basics," 2022. [Online]. Available: <https://moodle.uwtsd.ac.uk/mod/resource/view.php?id=321686>. [Accessed 18 December 2022].
- [2] G. Thomas, "Moodle, UWTSD if, elif, else," 2022. [Online]. Available: <https://moodle.uwtsd.ac.uk/mod/resource/view.php?id=330779>. [Accessed 19 December 2023].
- [3] G. Thomas, "Moodle UWTSD," 2022. [Online]. Available: <https://moodle.uwtsd.ac.uk/mod/resource/view.php?id=334930>. [Accessed 30 December 2022].
- [4] G. Thomas, "Moodle UWTSD," 2022. [Online]. Available: <https://moodle.uwtsd.ac.uk/mod/resource/view.php?id=336482>. [Accessed 30 December 2022].

- [5] Unknown, "Bing Calculator," [Online]. Available:  
<https://www.bing.com/search?q=what+is+pi&cvid=dc6df1ad18bc436bb6e713630ec74acc&aqs=edge.0.69i59j0l8.1391j0j9&FORM=ANAB01&PC=HCTS>. [Accessed 11 January 2023].
- [6] G. Thomas, "Moodle UWTSd," 2022. [Online]. Available:  
<https://moodle.uwtsd.ac.uk/mod/resource/view.php?id=336526>. [Accessed 12 January 2023].
- [7] 2022 CyberNews – Latest Cybersecurity and Tech News, Research & Analysis, November 2022. [Online]. Available: [https://uk.cybernews.com/lp/best-antivirus-software-uk/?utm\\_source=bing&utm\\_medium=cpc&utm\\_campaign=Bing\\_CN\\_AV\\_S\\_T1\\_UK\\_G\\_B\\_Desktop\\_English&utm\\_content=82257541801711&utm\\_term=antivirus&campaignId=429564565&adgroupId=1316116893366019&adId=82257541801711&ta](https://uk.cybernews.com/lp/best-antivirus-software-uk/?utm_source=bing&utm_medium=cpc&utm_campaign=Bing_CN_AV_S_T1_UK_G_B_Desktop_English&utm_content=82257541801711&utm_term=antivirus&campaignId=429564565&adgroupId=1316116893366019&adId=82257541801711&ta). [Accessed 06 November 2022].

FINAL GRADE

45 / 100

GENERAL COMMENTS

### Instructor

#### PART A

**A1** Variables - A good description of the variables and you've addressed part of the typing. The point with typing in Python is that it's dynamic and therefore it will infer the type, and the type can change during the lifetime of the program (you've correctly written that you can force the type too)

4/5

**A2** If/Else - I am happy with your explanation and code example 5 /5

**A3** Loops - You haven't quite said what a While loop is (ie repeats until the condition is met and number of iterations not known in advance. However, the code example is fine. For loops is OK

4/5

#### PART B

##### **B1 Variables**

Logical Stages/Testing - No logical stages presented or testing presented for any of the exercises? I think perhaps you missed this on the assignment brief. There's also a document "How to layout your assignment Part B" on Moodle.

0/5

Code - Good overall - you could have used a while loop for the age input.

9/10

## **B2 Shape**

Logical Stages/Testing - See comment above

/5

Code - A good effort - the algorithm for comparing the square and the circle will not work for all areas, it might be better to compare the relative lengths on the square/radius instead. However, a good effort. You could also have validated the input, so that it is not a minus number.

11/15

## **B3 Lists**

Not attempted?

## **B4 Turtle**

Code - A good attempt, you could have made more efficient by looping the Os (or using a function) and there is also potentially rotational symmetry on the Z

12/20

### **General comments:**

It is good practice to number the pages of your assignment to make it easier for the reader to navigate the document.

You should not use "you" in an academic assignment. This is what we call the active voice,

you need to use the passive voice. For example  
"You are able to force python to see the variable". -  
> "Python can be forced". It is a style for academic  
reports.

**OVERALL TOTAL FOR MODULE:** 45% - What you've  
handed in is good, but be careful to read the  
assignment carefully so you don't miss out on marks.

---

PAGE 1

---

PAGE 2

---

PAGE 3

---

PAGE 4

---

PAGE 5

---

PAGE 6

---

PAGE 7

---

PAGE 8

---

PAGE 9

---

PAGE 10

---

PAGE 11

---