Software Development 1
Conditional Statements 2

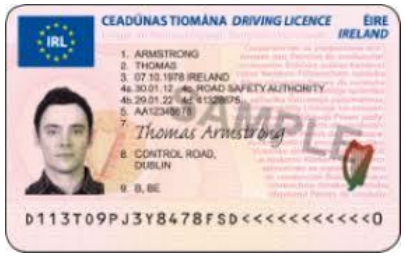
TUDublin –Tallaght Campus

E.Costelloe

Nested Conditional Statements

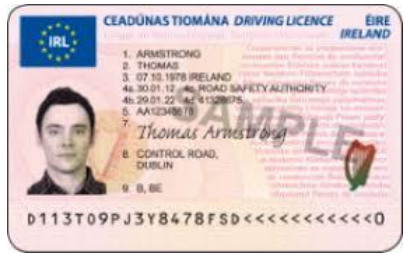
- We can now answer questions in one dimension.
- For example, the driver's license problem from the last lecture.
- But in the real world, questions may have more than one dimension, and our current conditional statements may not be able to solve it!
- Let's re look at the driver's license problem and add another dimension of complexity.

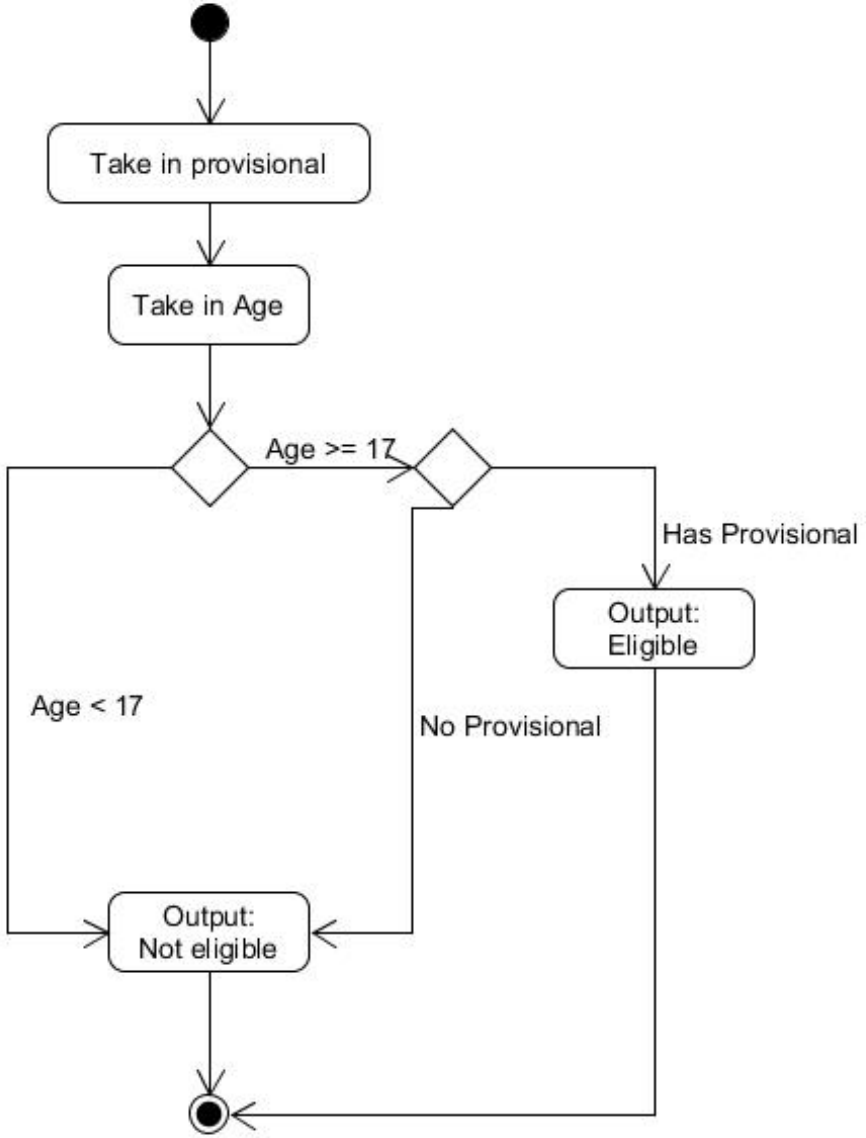
Nested Conditional Statements



- Driving license (car)
- Let's build a program that asks for a person's age and if they have already obtained a provisional license (A requirement to apply for a full license)
- Base on their age and Irish law and if they currently hold a provisional license, the computer will tell them if they are eligible to apply for a license or not.
- What would this algorithm look like?

Nested Conditional Statements





Nested Conditional Statements

- Step one of development:

```
age = int(input("Please enter your age:"))
has_provisional = input("Do you have a provisional license (y/n):")

if age >= 17:
    pass                        # pass is just a place holder.
else:
    print("You are not eligible for a driving license.")
```

- **pass** is a placeholder that you may use when a statement is required but you do not want any command or code to be executed - it is a null operation; nothing happens when it executes

Nested Conditional Statements

- Step two of development (solve one part first, then build):

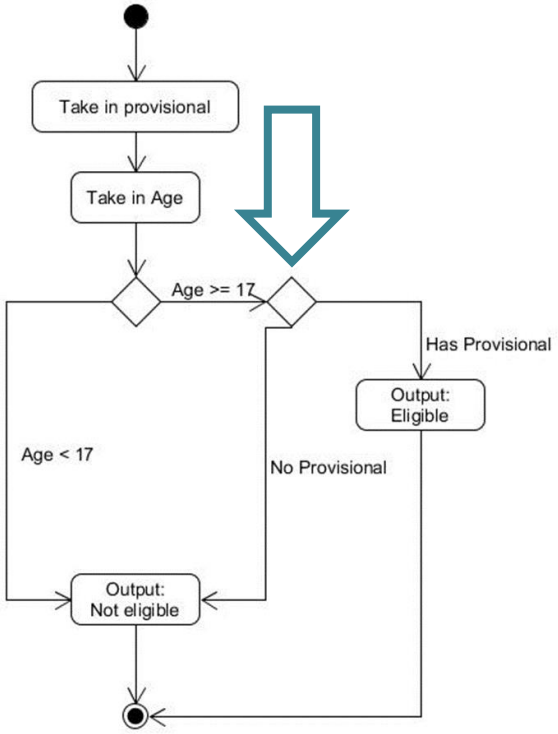
```
age = int(input("Please enter your age:"))
has_provisional = input("Do you have a provisional license (y/n):")

if age >= 17:
    print("You are eligible for a driving license.")
else:
    print("You are not eligible for a driving license.")
```

- Even though this is not the solution, it is a working iterative complete step towards it.
- Test this and then move on to next step.

Nested Conditional Statements

- Step three of development where does the second question (conditional) go?




```
age = int(input("Please enter your age:"))
has_provisional = input("Do you have a provisional license (y/n):")

if age >= 17:
    print("You are eligible for a driving license.")
else:
    print("You are not eligible for a driving license.")
```

Nested Conditional Statements

- Step three, the second conditional:

```
age = int(input("Please enter your age:"))
has_provisional = input("Do you have a provisional license (y/n):")

if age >= 17:
    if has_provisional == "y":
        print("You are eligible for a driving license.")
else:
    print("You are not eligible for a driving license.")
```

- Hint, follow your algorithm, the second conditional is only executed if the first conditional is evaluated to `True`, thus it must be inside the `if age >= 17:`
- *But there is something incorrect with this!*

Nested Conditional Statements

- Step three, the second conditional:

```
age = int(input("Please enter your age:"))
has_provisional = input("Do you have a provisional license (y/n):")

if age >= 17:
    if has_provisional == "y":
        print("You are eligible for a driving license.")
else:
    print("You are not eligible for a driving license.")
```

- What would the output be if: age = 19, `has_provisional` = “n” ? (and why?)

Nested Conditional Statements

- Final Solution:

```
age = int(input("Please enter your age:"))
has_provisional = input("Do you have a provisional license (y/n):")

if age >= 17:
    if has_provisional == "y":
        print("You are eligible for a driving license.")
    else:
        print("You are not eligible for a driving license.")
else:
    print("You are not eligible for a driving license.")
```

Nested Conditional Statements

- Caution with indentation at this point:

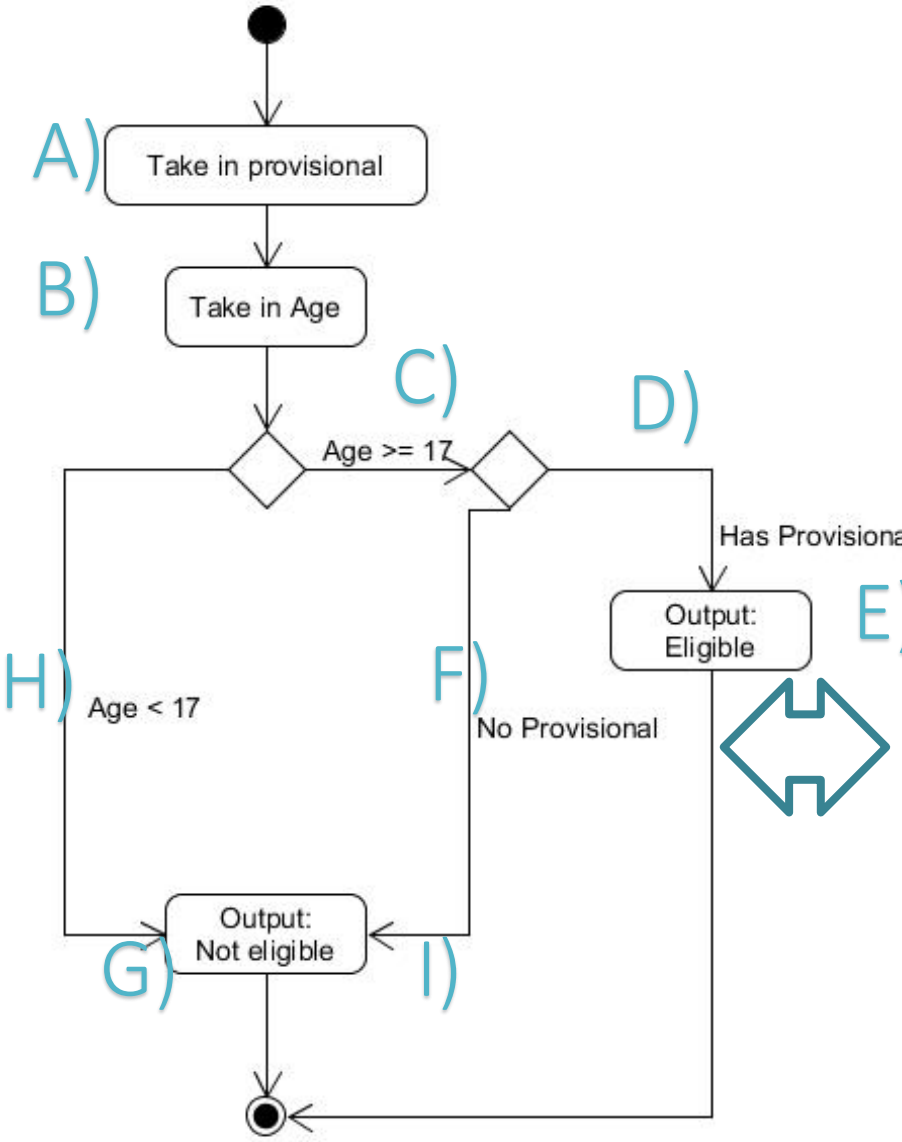
```
age = int(input("Please enter your age:"))
has_provisional = input("Do you have a provisional license (y/n):")

if age >= 17:
    if has_provisional == "y":
        print("You are eligible for a driving license.")
    else:
        print("You are not eligible for a driving license.")
else:
    print("You are not eligible for a driving license.")
```

- Test using

age	has_provisional
16	n
17	y
18	n

Nested Conditional Statements




```
A) has_provisional = input("Do you have a provisional license (y/n):")
B) age = int(input("Please enter your age:"))

C) if age >= 17:
D)       if has_provisional == "y":
E)                print("You are eligible for a driving license.")
F)       else:
G)                print("You are not eligible for a driving license.")
H) else:
I)       print("You are not eligible for a driving license.")
```

Class example:

- Write a python program that takes in two variables from a student, an age and a test score.

- This is the criteria:

  If the score is greater than or equal to 80 and the student is 16 or under, output: `Excellent`

  If the score's greater than or equal to 80 and the student is over 16, output: `Good`

  if the student score's less than 80, output: `Please try harder next time`

Nested Conditional Statements

- Solution:

```
age = int(input("Please enter your age:"))
result = int(input("Please enter your exam result:"))

if result >= 80:
    if age <= 16:
        print("Excellent")
    else:
        print("Good")
else:
    print("Please try harder next time")
```

Class example 2:

- Write a python program that takes in two variables from a customer, the number of products purchased and if they are a gold customer and outputs the total bill. Each product costs €57.23

- The criteria are:
- if they are a **gold customer,** they receive a discount of 3.5%, plus the following discounts:

Order Volume	% Discount of total cost
>= 25	5%
>= 100	10%

- Non gold customers receive **no** discounts
- The program must display the type of discount combination that the customer received, for example: “Your total is:€ 286.15 (No discount was applied to this order.)”

```
# constant variables
COST = 57.23
# Inputs
qty = int(input("Please enter the order quantity:"))
gold_customer = input("Are you a gold customer? (y,n):")
# convert gold customer to boolean (example purposes only)
if gold_customer == "y":
    gold_customer = True
else:
    gold_customer = False

if gold_customer:                    # Same as: if gold_customer == True:
    if qty < 25:
        total = (COST * qty) * (1 - .035)    # 0.965
        print("Your total is:€", round(total, 2),"(Including gold discount of 3.5%,no quantity discount was applied.)")

    elif qty < 100:
        total = (COST * qty) * (1 - (.050 + .035))    # .915
        print("Your total is:€", round(total, 2), "(Including gold discount of 3.5%, and a qty discount of 5%.)")

    else:
        total = (COST * qty) * (1 - (.035 + .10))    # 0.865
        print("Your total is:€", round(total, 2), "(Including gold discount of 3.5%, and a qty discount of 10%.)")

else:
        total = COST * qty
        print("Your total is:€", round(total, 2), "(No discount was applied to this order.)")
```

Complete following program so that it will use nested ifs to determine if x and y are equal to 30 and 10 respectively and print a message to that effect. Print a different message if x = 30 but y not = 10. If x is not equal to 30 a message to that effect should be printed.

```
x = 30
y = 10

if x == 30:
    if y == 10:
        print("Both x = 30 and y = 10")
    else:
        print(" x = 30 but y is not = 10")
else:
    print("x is not = 30")
```

Write the program to test whether an integer read in from the user is a single digit (<10) and odd . Assume the number is greater than or equal to 0.

```
number = int(input("Enter a number >= 0 :"))
if number < 10:
    if number % 2 != 0:
        print("The number, a single digit", number, "is odd")
    else:
        print("The number, a single digit", number, "is even")
else:
    print("The number", number, "is not a single digit")
```

How would one know if a number is divisible by 4?,by 100?

if number % 4 == 0:

if number % 100 == 0:

Doing Math on User Inputs

If we wish to perform mathematical calculations on user inputs, see the following:

If we enter the following, we generate a run time error as shown

```
number = input("Enter a number >= 0 :")
result = number ** 2
print(number, "squared is ", result)
```

Run-time error:

```
Enter a number >= 0 :q
Traceback (most recent call last):
  File "C:/Users/ecostelloe/PycharmProjects/week4/lectureNotes.py", line 85, in <module>
    result = number ** 2
TypeError: unsupported operand type(s) for ** or pow(): 'str' and 'int'
```

This won't work because Python won't convert types in expressions unless they are all numeric, and inputs from a user is always returned to(input to) the program as a string.

isdigit() string method

In Python, **isdigit()** is a built-in method used for string handling.

The isdigit() methods returns “True” if all characters in the string are digits, Otherwise, It returns “False”.

This function is used to check if the string contains digits such as : **0123456789**

It doesn’t take any arguments, i.e. No values in the parentheses

Fractions are not considered to be digits

stringname.isdigit() will return True or False as the case may be

Input Validation

Solution: Use the isdigit() string method to test if the value entered is a digit

```
number = input("Enter a number >= 0: ")
if number.isdigit():
    result = number ** 2
    print(number, "squared is ", result)
else:
    print(number, "cannot be squared as it is not a digit ")
```

Now the program is more robust and will deal with invalid user input.

Output:

```
Enter a number >= 0: q
q cannot be squared as it is not a digit
```

NB ::What still needs to be added to the code??

```
result = int(number) ** 2
```

Class example :

Mopeds R' Us rents mopeds at the beach. To promote business during slow weekdays, the store gives a huge discount. The rental charges are as follows

Moped Type	Weekday Rental	Weekend Rental
50cc Mopette	€15.00 for the first 3 hours, €2.50 per hour after that	€30.00 for the first 3 hours, €7.50 per hour after that
250cc Mohawk	€25.00 for the first 3 hours, €3.50 per hour after that	€35.00 for the first 3 hours, €8.50 per hour after that

Design and write a program that computes the rental charge, given the type of moped, when it is rented (weekend or weekday) and the number of hours rented, all three pieces of information to be input from the user.

Data

Constants:

DAY_SMALL_MOPETTE = 15.00 :

END_SMALL_MOPETTE = 30.00:

ADD_DAY_SMALL = 2.50:

ADD_END_SMALL = 7.50:

FEE_HOURS = 3:int

Additional constants needed here for 250cc moped

Inputs:

typem : String

day : String

hours :int

Outputs:

fee : float

Other:

extra_hours: int

PseudoCode

Input Data

Output (Enter mopette type (50cc or 250cc):)

Input typem

Output (Enter which part of the week rented (weekend or weekday):)

Input day

Output(Enter amount of hours used(number greater than zero):)

Input hours

Calculate hours over 3

extra_hours = hours - FEE_HOURS

Calculate Fee For Small mopette

```
if  typem =  "50cc"
   #  check for weekday
    if day = "weekday"
        fee = DAY_SMALL_MOPETTE
        if extra_hours > 0
          fee =  fee +  extra_hours * ADD_DAY_SMALL
    # weekend
    else
      fee = END_SMALL_MOPETTE
      if extra_hours > 0
          fee = fee + extra_hours * ADD_END_SMALL
```

large mopette

```
else
  #  weekday
   if day = "weekday"
       fee = DAY_LARGE_MOP
       if extra_hours > 0 then
           fee = fee +  extra_hours * ADD_DAY_LARGE
   else     #  weekend for large mopette
      fee = END_LARGE_MOP
      if extra_hours > 0
         fee = fee +  extra_hours * ADD_END_LARGE


Output(fee,  typem, day, hours)
```

Logical operators

- Next we will look at logical operators, **not**, **and**, **or** (precedence in the order of not, and, or)

- Conditional statements use logical operators for complex questions (with two or more parts): if you go to the shop **and** you have money you can buy chocolate.

- Similar in idea to nested conditional statements, but can add even more dimensions, without the need to add additional nested conditional statements.

- A simple condition is where two values are involved. A compound condition is a condition that combines simple conditions with logical operators, allowing you to make decisions based on how multiple groups of values compare.

logical operators: OR

A compound condition created with an **or** is True as long as at least one of the simpler conditions is True

Conditional Statement 1	Conditional Statement 2	Overall evaluation
True	True	True
True	False	True
False	True	True
False	False	False

```
x = 10
y = 14

if x == 10 or y > 6:
    print("T or T ")
if x == 10 or y < 6:
    print("T or F ")
if x == 20 or y > 14:
    print("F or F ")
```

Output

T or T

T or F

logical operators: AND

A compound condition created with an **and** is True only if all of the conditions are True

Conditional Statement 1	Conditional Statement 2	Overall evaluation
True	True	True
True	False	False
False	True	False
False	False	False

```
x = 10
y = 14
if x == 10 and y > 7:
    print("T and T ")
if x == 10 and y > 14:
    print("T and F ")
if x == 20 and y > 17:
    print("F and F ")
```

Output

T and T

logical operators: NOT

Putting **not** in front of a condition creates a new condition that evaluates to the opposite of the original

Conditional Statement 1	Overall evaluation
True	False
False	True

```
x = 10
if not x == 10:   # equivalent to x != 10
    print("x not equal to 10")
else:
    print("x equal to 10")
if not x < 10:
    print("x not less than 10")
```

Output

x equal to 10

x not less than 10

Example: OR

- Take in two variables, car year and fuel type

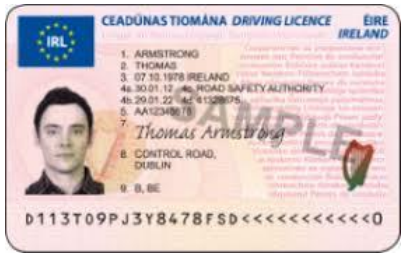
- Criteria:

  If the car is ten years old or more, or it is a petrol, it is classified as a high pollution car, otherwise, it is a low pollution car.

```
car_age = int(input("Please enter the age of the car:"))
car_fuel = input("Please enter the cars fuel type (petrol/diesel):")

if car_age >= 10 or car_fuel == "petrol":
    print("Classification: High Pollution.")
else:
    print("Classification: Low Pollution")
```

Example: AND



- Driving license (car)
- Lets build a program that asks for a persons age and if they have already obtained a provisional license (A requirement to apply for a full license)
- Base on their age and Irish law and if they currently hold a provisional license, the computer will tell them if they are eligible to apply for a license or not.
- What would this algorithm look like?

Example: AND

- Solution:

```
age = int(input("Please enter your age:"))
has_provisional = input("Do you have a provisional license (y/n):")

if age >= 17 and has_provisional == "y":
    print("You are eligible for a driving license.")

else:
    print("You are not eligible for a driving license.")
```

Example: AND

- Further solution:

```
age = int(input("Please enter your age:"))
has_provisional = input("Do you have a provisional license (y/n):")

if age >= 17 and has_provisional == "y":
    print("You are eligible for a driving license.")

elif age >= 17 and has_provisional != "y":
    print("You are old enough to apply for a license but you
           need a provisional license.")
else:
    print("You are not eligible for a driving license as you are not the correct age
           to apply, you need to be 17 years of age or older.")
```

Class example: part 1

- Confirm that an email address is the correct format:

- Lets assume that an email address needs the following:

  @ symbol

  minimum 6 characters in length

  contains at least one “.”

Example: part 1

```
email = input("Please enter your email address:")
if "@" in email and "." in email and len(email) >= 6:
    print("valid email address.....")
else:
    print("email address is not valid...")
```

Class example: part 2

- Confirm that an email address is correct format:

- Lets assume that an email address needs the following:

    @ symbol

    minimum 6 characters in length

    contains at least one “.”

Added:

    The “@” symbol must be before the “.”

Example: part 2

```
email = input("Please enter your email address:")
if "@" in email and "." in email and len(email) >= 6:
    if email.index("@") < email.index("."):
        print("valid email address.....")
    else:
        print("email address is not valid...")
else:
    print("email address is not valid...")
```

joe@amail.com

0 1 2 3 4 5 6789…

Class example:

- If a member wishes to login to their network they must enter their username and a password that are recognised together. E.G if John Smith wants to log in he has to enter **J.Smith** for his username and **python** for his password. If he doesn't enter both exactly that way he cannot login.
- Write the code that would read and check the login combination and print **Access granted** or **Login failed** as appropriate

```
username = input("Enter your username: ")
password = input("Enter your password: ")

if username == "J.Smith" and password == "python":
    print("Access granted")
else:
    print("Login failed")
```

Conditional Statements for Menu's

- Many programs have menus to select a specific task from many options.

- It is unreasonable to expect a user to open a program to do each of the different tasks.

- Write a program that has a menu for two specific tasks:
    - Calculate the area of a circle
    - Calculate the volume of a sphere

Modules

Modules are files that contain code meant to be used in other programs

They usually group together a collection of programming related to one area, they are like toolkits that you can use when you need them

To use the math module in your program code the following as the first line of code:

```
import math
```

Importing the math module allows you to use a number of mathematical functions and constants e.g. pi

The constant pi is accessed by using **`math.pi,`** you can access the constant from an imported module by giving the module name, **math**, followed by a dot, followed by the constant name. This method of access is called dot notation.

Example: Menu

- Solution:

```
import math

# Menu options
print("\t\t*************************")
print("\t\t*       My Program      *")
print("\t\t*************************")
print("\t\t*1) Area of a circle    *")
print("\t\t*2) Volume of a sphere  *")
print("\t\t*************************")
option = int(input("\t\tPlease enter option:"))

print("\n\n")

if option == 1:
    radius = float(input("Please enter circle radius:"))
    area = round(math.pi * (radius ** 2), 4)
    print("The area of circle with radius", radius, "is", area)
else:
    radius = float(input("Please enter sphere radius:"))
    volume = round((4/3) * math.pi * (radius ** 3), 4)
    print("The volume of sphere of radius", radius, "is", volume)
```

Example: Menu with invalid entry trapped

- Solution:

```
if option == 1:
    radius = float(input("Please enter circle radius:"))
    area = round(math.pi * (radius ** 2), 4)
    print("The area of circle with radius", radius, "is", area)
elif option == 2:
    radius = float(input("Please enter sphere radius:"))
    volume = round((4 / 3) * math.pi * (radius ** 3), 4)
    print("The volume of sphere of radius", radius, "is", volume)
else:
    print("Invalid option")
```