STEM

OPPORTUNITIES IN AVIATION & AEROSPACE

Welcome to the Civil Aviation Authority Duke of Edinburgh's Award Skills Challenge

An introduction and how we expect this to be answered

In this DofE task there will be 2 styles of answering. Those for written questions will have the below text box for you to edit and answer.

For Example:

Question: What is your favourite Ice Cream?

Answer: Chocolate Moose

For coding answers we will ask you to output your answers using Python's in built Print function. This will take the text put inside of it's brackets and write it onto the page. Try it by hitting the play button in the next box. Feel free to have a play around with what's written in the box!

```
print("Hello World")
```

Chapter 1: CIA Triad

CIA TRIAD

The CIA triad is a fundamental part of cyber security and so is where we'll start this activity. Your first assignment is to understand it in order to answer the questions below. Below we have provided some handy links you can use, but feel free to research beyond them.

Source 1 Source 2

Question: What does the CIA triad stand for?

Answer:

Question: How is the CIA triad used in cyber security?

Answer:

Question: What is an example of a technology or technique that helps enforce each of the triad areas?

Answer:

Chapter 2: Online Threats

Explain the most common types of online threat and how to protect against them.

We will specifically look at:

- Phishing: Source 1 Source 2

- Malware: Source 3 Source 4
- Password Cracking: Source 5 Source 6

National Cyber Security Centre (NCSC) Guidance

Explanation of the 3 attack types and way to protect against in a 2000 character document.

Feel free to use the above sources, or do your own research.

Answer:

Question: What are phishing, malware, and password cracking and how can you protect against them? (Maximum 2000 characters)

Answer:

Question: Using the NCSC guidance, create a memorable password.

Answer:

Chapter 3: Cloud Systems

Understand what it meant by Cloud systems on the internet. Difference between internet and intranet. How is the cloud utilised in internet vs intranet. 2000 Characters limit

Source 1

Source 2

Above we have provided some handy links you can use, but feel free to research beyond them

Answer:

Chapter 4: Cryptography

Python Basics

Integers, Floats, and Numberic Operations

In Python there are different types of numbers, the ones we'll be focusing on for this exercise are **integers** and **floats**.

- **Integers** are whole numbers, whether zero, negitive, or positive. Examples include:
  - 20, 0, -5, 1206, -145863912
- **Floats** are numbers that feature a decimal point. Examples include:
  - 0.05, -12.6, 5.0, 55593.2213

It is possible to convert between integers and floats using casting:

```
(int) (5.005)
(float) (27)
```

Python also allows you to perform the main 4 mathematical operations and even some additional ones such as:

- **+** which adds 2 numbers

```
Syntax: 19 + 3
```

- ***** which multiplies 2 numbers

```
Syntax: 5 * 19
```

- **-** which minuses 2 numbers

```
Syntax: 22 - 5
```

- **** which divides 2 numbers

```
Syntax: 18 \ 6
```

- **%** which finds the modulus of 2 numbers, this is the same as the remainder of a division

```
Syntax: 29 % 6
```

Try these operators out for yourself

Question: By default, using integers, which of the above operations results in an integer result and which result in a float result? (You may use the above cell to write the operations and see the results)

Answer:

Question: Using the below cell, compute and print 129 minus 16 divided by 2 multipled by 9 modulus 27

Question: What is the order of operations Python follows?

Answer:

Strings, Characters, & Variables

In Python words and sentences are stored as a **string** which is an array of **characters**. Strings are formed by using quotation marks around a word or sentence, like the following:

```
'hello' or "hello"
'this is a sentence' or "this is a sentence"
```

Question: In the below cell print the following sentence:

The quick brown fox jumps right over the lazy dog

Strings in Python are an array of letters, so we can call each letter individually by using the following syntax. Remember in computer science we count from 0!

```
"word"[0]
"sentence"[5]
```

Question: Using the above syntax, print the 2nd letter in the word *quack*, and then print the 5th letter in the word *souvenir*.

The **chr()** function takes in an integer and finds the letter, number, or symbol that exists on unicode table. The most common section of this table, and the one we'll be using for this exercise is **ASCII** which ranges from 32 to 126. You can use the chr function as follows:

```
chr(43)
```

Question: What is character 63 when printed on Python?

Characters can also be strung together for form a String, this can be done by using the mathmatical symbol **+**. For example, if we want to get the word *and* we can use the following syntax:

```
chr(97) + chr(110) + chr(100)
```

Question: What is the name of the film written below? Print it out in the below cell.

65, 105, 114, 112, 108, 97, 110, 101, 33

Alternatively, you can add to an already existing string either with an additional character or with a string of characters.

```
"puzzle" + chr(115)
"cup" + "board"
```

```
print("puzzle" + chr(115))
print("cup" + "board")
```

Since we can convert **integers** to **characters** via the ASCII lookup table, we can do the same in reverse by using the **ord** function. For example:

```
ord('a')
```

will return 97.

Question: Using the ord function, find the integers that represent the following sentence (including space, capital letters, and exclamation mark):

Zero Hour!

One of the Python functions you'll be using in the later is the length function, which as the name suggests returns an **integer** of the length of a **String** like the following:

```
len("xylophone")
```

Question: Using the code box below find the length of word for the fear of long words.

Variables in Python are simply containers for storing data values. However, unlike other languages, Python has no command for declaring a variable! Variables can store any data types, such as Boolean, Integers, Floats, and Strings. Variables can also be used to combine strings or perform arithmetic, and can also be redefined later down the line. Where

```
x = "hello"
```

in one instance, it could be reassigned to

```
x = 5
```

Question: Using the box below, try defining two variables. Define one as `hello` and the other as `world`, and then combine them! (Don't forget the space!)

Logic Gates & If Statements

In Python there are different ways to direct the computer through your code one of the most common ways is with **if**, **else** and **elif** gates. These require **booleans** which are true or false statements. These use common symbols such as:

```
> meaning greater than
< meaning less than
== meaning equal to
```

These can even include equations on each side, so

Question: Using the below cell find out what is the result of this equation?

```
11 * 17 == 9 * 21
```

if this is false rewrite the equation with the correct symbol below it

---

Statements

The **if** gate directs the computer to a set of code if the boolean is **true**, otherwise it skips it if it's **false**.

The **elif** (else if) gate directs the computer to a set of code if the **if** boolean was **false** and the **elif** boolean is true.

The **else** gate directs the computer to a set of code if the above **if** boolean and **elif** boolean/s were **false**.

```
<code>

if boolean:
  <code that runs if boolean is true>

elif boolean:
```

```
    <code that runs if the elif boolean is true and the if boolean is false>

  else:
    <code that runs if the if and elif statements are false>

  <code>
```

```
if 5 + 5 == 10:
  print("The if statement is true!")

elif 729 / 9 == 27:
  print("The if statement is false and the elif statement is true!")

else:
  print("The if and elif statement are false!")
```

For Loops & Range Function

For Loops

For loops in python are used for iterating over a sequence (lists, tuples, dictionaries, sets, or strings). With a For Loop, we can execute a set of statements once for each item in a list.

For example, to print each item in a list:

```
aircraft = ["A380", "A320", "747"]
for x in aircraft:
  print(x)
```

They can also be used to loop through the letters of strings. For example, to print each letter in the word "aeroplane":

```
for x in "aeroplane":
  print(x)
```

Through the addition of an `if` and `break` statement, we can exit the loop before it has finished going through all the items.

For example:

```
aircraft = ["A380", "A320", "747"]
for x in aircraft:
  print(x)
  if x == "A320":
    break
```

The `break` statement can also be used to exit the loop before the `print` command:

```
aircraft = ["A380", "A320", "747"]
for x in aircraft:
  if x == "A320":
    break
  print(x)
```

Question: Create a for loop, and print out each letter in the word `"Bombardier"`, stopping the loop at the letter `d`

Functions

A function is simply a block of code that will run when it is called. It can be defined however you like. You can pass parameters into a function, which can also return data as a result.

Functions in Python are defined using the `def` keyword, and called by using the function name followed by parenthesis.

For example:

```
def printing():
  print("This is printed by calling the function!")

printing()
```

Information can be passed into a function using what are known as arguments. These arguments are specificed inside the parenthesis after the function name. You can utilise as many arguments as you like, just remember to separate them with a comma.

For example:

```
def Add_Type(model):
  print("Airbus " + model)

Add_Type("A320")
Add_Type("A380")
```

Question: Try defining the function `shuttlename` with the argument `shuttle`, and use it to add each shuttle name (Discovery, Atlantis, Challenger) onto the phrase `"Space Shuttle "`.

Range Function

The `range()` function returns a sequence of numbers which start from 0 by defaukt. They increment by 1 each time (by default) and will stop before a specified number.

The syntax used is `range(start, stop, step)`. This allows you to define each parameter as you require it. The `start` and `step` parameters are optional and will default to `0` and `1` respectively.

`start` - Defines which integer to start at.

`stop` - Defines which integer to stop at. (Remember, it will stop at this number, so if you want to stop at `10`, you should input `11`)

`step` - Defines the incrementation at which to increase by.

For loops are utilised with the range function to loop through the range specified.

For example, to create a sequence from 5 to 10:

```
x = range(5, 11)
for n in x:
  print(n)
```

Question: Try using the range function to output the 5 times table, stopping at `50`.

Python Exercise

Using the examples and what you have learnt above about the different types of operators and functions in Python, you can have a go at writing Python code for some of the commonly known encryption methods.

Caesar Cipher

Research and give a brief summary abot Caesar Cipher (3000 characters limit).

You may use the below source or do your own research.

[Source 1]

Answer:

The Caesar Cipher should go through the following processes:

- Take in the **plain text** string and **shift** integer from the user.
- Run through each character in the **plain text**,
    - Shift any *lower* case character by the **shift** value. Use the ASCII Table to find out the lower case values. Watch out for any overflows!
    - Shift any *upper* case characters by the **shift** value. Use the ASCII Table to find out the lower case values. Watch out for any overflows!
    - Optional: Shift any numbers and/or special characters.
- Print out the shifted **plain text** string.

Exercise

Using the # comment function in the below code, explain the different functions and what exactly are they doing for operations of this cipher code.

```
#
ciphertext = ""

#
plaintext = input("Enter the plaintext: ")

#
shift = int(input("Enter the shift value: "))


#
for char in plaintext:

  #
  if char.isalpha():

      #
      is_uppercase = char.isupper()

      #
      char_code = ord(char)

      #
      shifted_char_code = (char_code - ord('A') + shift) % 26

      #
      shifted_char = chr(shifted_char_code + ord('A'))

      #
      if is_uppercase:
        shifted_char = shifted_char.upper()
      else:
        shifted_char = shifted_char.lower()

      #
      ciphertext += shifted_char

  #
  else:
    ciphertext += char


#
print("The ciphertext is:", ciphertext)
```

Atbash Cipher

Explain the history of the Atbash Cipher using 500 words.

Answer:

Using the below Atbash Cipher process description, write the code to run the cipher for ***Secret Message***

The Atbash Cipher code should go through the following processes:

Step 1: Create the alphabet and the reversed alphabet

Step 2: Define the function for the Atbash cipher

Step 3: Loop through each letter in the input text

Step 4: Check if the letter is in the alphabet

- Find the position of the letter in the alphabet
- Get the corresponding letter from the reversed alphabet
- Preserve the original case (uppercase/lowercase)
- Add the new letter to the result
- If the character isn't a letter, just add it to the result

Step 5: Print the plaintext and ciphertext results