Level

2

Secret Codes

Keep track of your progress by
ticking off the boxes below:

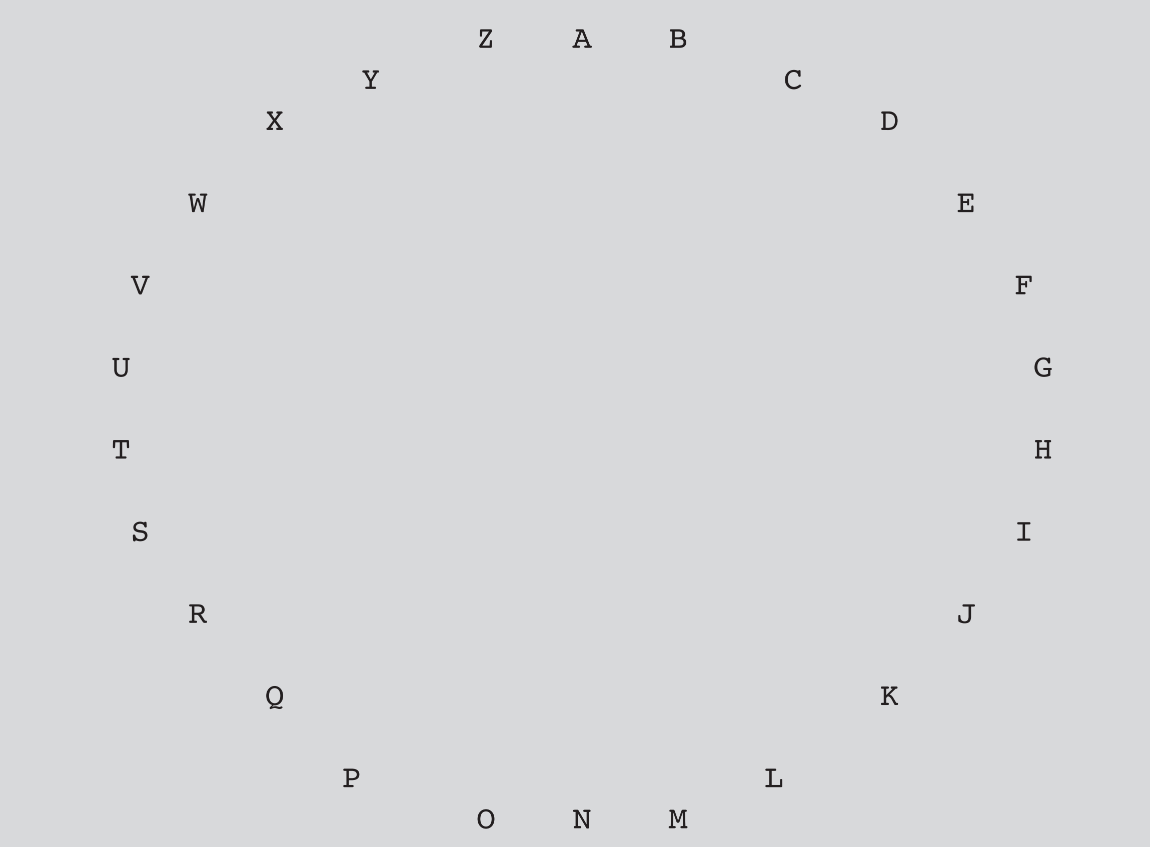
Introduction

Put your turtles away, today we're going to learn how to send secret messages!

The Code Club Cipher

A cipher is a type of secret code, where you swap the letters around so no-one can read your message. We'll be doing one of the oldest and most famous ciphers, the Caesar Code. Named after a famous person, not Dog food.

We start by drawing the letters in a circle.



To make a secret letter from a normal one, we need to have a secret key. Let's randomly use three:

```
    A + 3 = D        T + 3 = W        Z + 3 = C
```

We start at A, and count forward 3 letters: B, C, D. So the letter A turns into the letter D. To decode, we do the same thing, but in reverse. We start at D and count backwards to get A.

Step 1: The Alphabet

Activity Checklist

1. First we'll have to tell python the alphabet. Open up IDLE, create and save a new file with the following code. □

```
alphabet = “abcdefghijklmnopqrstuvwxyz”
print(len(alphabet))
```

2. When you run it (from the menu), it should print 26. Make sure you've got all the letters in, or your secret code won't work. □

If you're happy with your alphabet, we can start to encode a letter. □

Step 2: Encoding a letter

Activity Checklist

Like on the wheel above, we find the position of the letter, we count forwards, and then use the letter we end up at.

1. Write in the code below and run it: □

```
alphabet = “abcdefghijklmnopqrstuvwxyz”
letter = “a”
secret = 3
pos = alphabet.find(letter)
newpos = (pos + secret)
if newpos >= 26:
    newpos = newpos - 26
secretletter = alphabet[newpos]
print(secretletter)
```

Keep track of your progress by ticking off the boxes below:

We look up where “a” occurs in the alphabet, add the secret number to count forwards, check to see if we’ve gone too far, and then look up the letter again.

2. Run the code, and see what happens. □

Let’s look at the code again, but take it slowly:

You don’t have to type this in!

```
# alphabet is the name for the text of a to z
alphabet = “abcdefghijklmnopqrstuvwxyz”

# the letter and secret we’ll use to encode it
letter = “a”
secret = 3

# find the position of the letter, and
# python will tell us a number from 0 to 25
pos = alphabet.find(letter)

# we count forward a few letters
newpos = (pos + secret)

# if we count too far, we’ll have to start
# from the beginning
if newpos >= 26:
    newpos = newpos - 26

# now we look up the letter at this new pos
secretletter = alphabet[newpos]

# and output it to the screen
print(secretletter)
```

There is a lot of python going on here, so don’t worry if it seems a bit too much to understand at first. Much of is quite like scratch. `if newpos >= 26` is just an `if` statement, a thing that only runs the code underneath if the condition is correct. It uses an indented block, just like `for` and `def` that we’ve seen earlier.

Now we can encode a letter, how about decoding one?

Step 3: Uncovering the letters

Like the code from the earlier step, we find the position of the letter, but we go backwards through the alphabet to decode.

Keep track of your progress by
ticking off the boxes below:

Activity Checklist

1. Try copying the following code in and running it! □

```
alphabet = “abcdefghijklmnopqrstuvwxyz”

secret =17
secretletter = “r”

pos = alphabet.find(secretletter)

newpos = pos - secret

if newpos < 0:
    newpos = newpos + 26

letter = alphabet[newpos]

print(letter)
```

Step 4: Building functions

Now we can wrap these bits of codes in functions. Take the first code and make it a function `encode`, and the second one in `decode`. We learned how to define new functions in lesson two, but we're going to use a special command in python, `return`.

Some functions just do things, other functions can calculate things. Although we've looked at passing in values to a function, we haven't looked at handing back a result. Using `return` we can say that the function is complete and what the result is. It is called `return` because we are returning a value.

Activity Checklist

1. Copy and paste the earlier code into a new file, and edit it to look like this: □

```
alphabet = “abcdefghijklmnopqrstuvwxyz”

def encode(letter, secret):
    pos = alphabet.find(letter)

    newpos = (pos + secret)

    if newpos >= 26:
        newpos = newpos - 26

    return alphabet[newpos]
```

```
def decode(letter, secret):
    pos = alphabet.find(letter)

    newpos = (pos - secret)

    if newpos < 0:
        newpos = newpos + 26

    return alphabet[newpos]

print(encode(“a”, 17))
print(decode(“r”, 17))
```

Remember you can use the Tab key in Idle to indent code (and selections too)

2. Try encoding and decoding some letters! □

Step 5: Sending secret words and back again

Now we have some functions, let’s use them to encode words. What we’re going to do is go through each letter in the word, and encode it if it’s in the alphabet (so we skip spaces and punctuation)

Activity Checklist

1. Underneath the new functions in the last step, write the following code in the same file: □

```
secret = 17
message = “hello world”

output = “”

for character in message:
    if character in alphabet:
        output = output +  encode(character, secret)
    else:
        output = output + character

print(output)

secret = 17
message = “yvccf nficu”
output = “”

for character in message:
    if character in alphabet:
        output = output + decode(character, secret)
    else:
        output = output + character
print(output)
```

2. Run it and see what happens. □

The first bit of code should print "**yvccf nficu**", which is the secret version of "**hello world**". The second bit decodes it back, and should print "**hello world**"

Step 6: Decode some secret messages

Here are some secret messages to try and decode!

`n frperg clguba`, the secret is 13.

Answer: ____

`yj tjp gdfz xjyz xgpw?`, the secret is 21.

Answer: ____

`axkxcb jan hxda oarnwmb`, the secret is 9.

Answer: ____

Things to Try

Why not try and send your own secret messages?