Level 3

Hangman

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

Introduction

Let's build a game: Hangman! The computer will pick a word, and the player can guess it letter-by-letter, but if they make too many wrong guesses, they'll lose.

Step 1: Pick a word

We start by picking a random word, so let's begin!

Activity Checklist

1. Open IDLE, and open a new window. ☐
2. Write in the following code: ☐

```
from random import choice
word = choice(["code", "club"])
print(word)
```

3. Save your program, and run it. What word does it print? ☐
4. Run it again, does it print a different word? ☐

Each time you run this program, it picks a random word from the list `["code", "club"]`, using the choice function.

Step 2: Guess a letter

Now we've picked a word, let's find out how to guess a letter.

Activity Checklist

1. With the same file, edit the code so it looks like this ☐

```
from random import choice
word = choice(["code", "club"])
out = ""
```

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

```
for letter in word:
    out = out + "_"
print("Guess a letter in the word:", out)
```

2. Save and run the program. ☐
3. You should see "Guess a letter in the word: ____", in the output window (the other window, not the one you've written your program in.) ☐

We use a for loop to build up some text with an underscore _ for each letter in the word. The word "code" put in, will write out ____ to the screen.

4. Let's guess a letter! Change the code to look like this ☐

```
from random import choice
word = choice(["code", "club"])
out = ""
for letter in word:
    out = out + "_"
print("Guess a letter in the word, then press enter:", out)
guess = input()
if guess in word:
    print("Yay")
else:
    print("Nope")
```

We use a new function `input()` to find out what the player typed. We use if to find out if the letter was in the word.

We've got the essentials down, so let's continue onward.

(Python 2 Note: Use `raw_input` if you're on an old version of python)

Step 3: Track the guesses

Now we're going to use two features of python, lists and the `while` loop.

Activity Checklist

1. In the same file, edit the code to look like this: ☐

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

```
from random import choice
word = choice(["code", "club"])
guessed = []
while True:
    out = ""
    for letter in word:
        if letter in guessed:
            out = out + letter
        else:
            out = out + "_"
    if out == word:
        print("You guessed", word)
        break

    print("Guess the word:", out)
    guess = input()
    if guess in guessed:
        print("Already guessed", guess)
    elif guess in word:
        print("Yay")
        guessed.append(guess)
    else:
        print("Nope")
    print()
```

2. Run the code, try guessing the letters. ☐

What we've done is put a loop, like `forever` in scratch, that will keep asking for letters from the player, until they guess the word.

We also use a list, `guessed`, which we add the letters to when they're right. This program will loop forever until all the letters are guessed.

Step 4: Track the mistakes

Hangman should only give you a few chances to guess, rather than trying every letter in turn.

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

Activity Checklist

1. Edit the existing file, and change it to look like the following: ☐

```
from random import choice

word = choice(["code", "club"])

guessed = []
wrong = []

while True:

    out = ""
    for letter in word:
        if letter in guessed:
            out = out + letter
        else:
            out = out + "_"

    if out == word:
        print("You guessed", word)
        break

    print("Guess the word:", out)

    guess = input()

    if guess in guessed or guess in wrong:
        print("Already guessed", guess)
    elif guess in word:
        print("Yay")
        guessed.append(guess)
    else:
        print("Nope")
        wrong.append(guess)

    print()
```

We're using a new list, `wrong`, to store all the guesses that weren't right

Only one last thing before the game is complete, which is to only have a few chances to guess.

Step 5: Only a few chances

Hangman

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

Activity Checklist

1. Edit the file, to introduce a new variable, tries:

```
from random import choice

word = choice(["code", "club"])

guessed = []
wrong = []

tries = 7

while tries > 0:

    out = ""
    for letter in word:
        if letter in guessed:
            out = out + letter
        else:
            out = out + "_"

    if out == word:
        break

    print("Guess the word:", out)
    print(tries, "chances left")

    guess = input()

    if guess in guessed or guess in wrong:
        print("Already guessed", guess)
    elif guess in word:
        print("Yay")
        guessed.append(guess)
    else:
        print("Nope")
        tries = tries - 1
        wrong.append(guess)

    print()

if tries:
    print("You guessed", word)
else:
    print("You didn't get", word)
```

2. Run the file, and see what happens when you guess wrong letters

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

Step 6: Add some new words in

Activity Checklist

1. Find this line in the source code: ☐

```
word = choice(["code", "club"])
```

2. Edit it to add more words. Why not try: ☐

```
word = choice(["code", "club", "robot", "party"])
```

Remember to put the words in quotes, and put a comma between them to make a list of words.