Stimulus: *Intrusion Testing*

WINDOWS XP STILL ALIVE

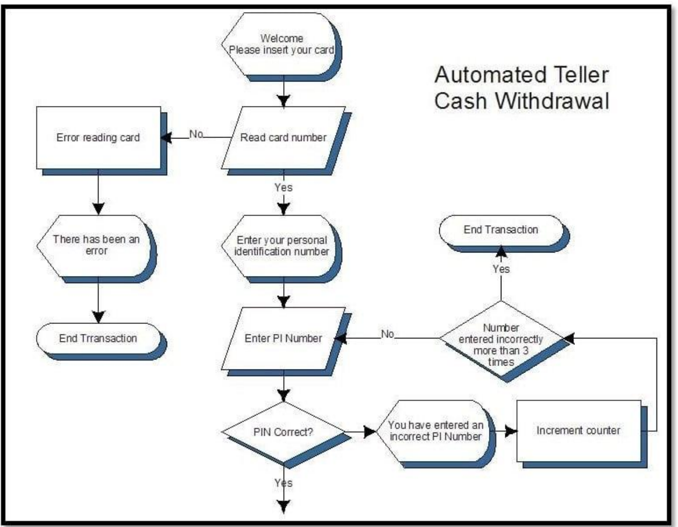
- Early 2014 – 95% of ATMs run on Windows XP
- Support killed off in April 2014
- Over 9000 vulnerabilities

RED TEAM

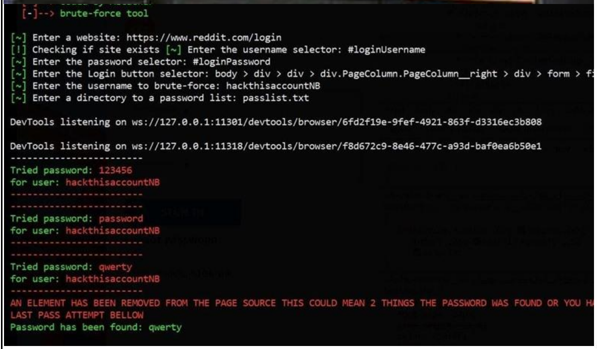
- Offensive Security
- Ethical Hacking
- Exploiting vulnerabilities
- Penetration Tests
- Black Box Testing
- Social Engineering
- Web App Scanning

BLUE TEAM

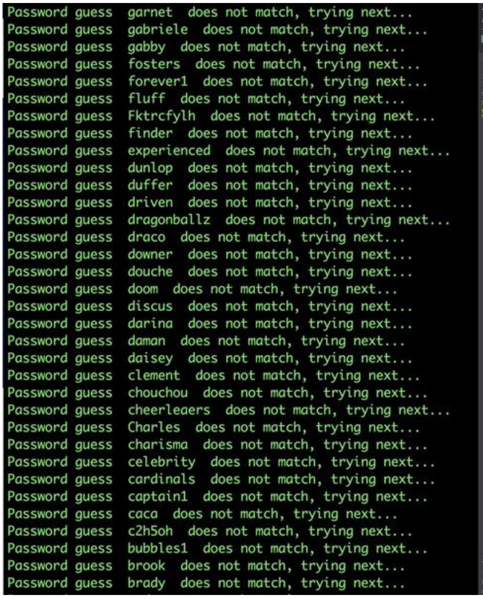
- Defensive Security
- Infrastructure protection
- Damage Control
- Incident Response(IR)
- Operational Security
- Threat Hunters
- Digital Forensics



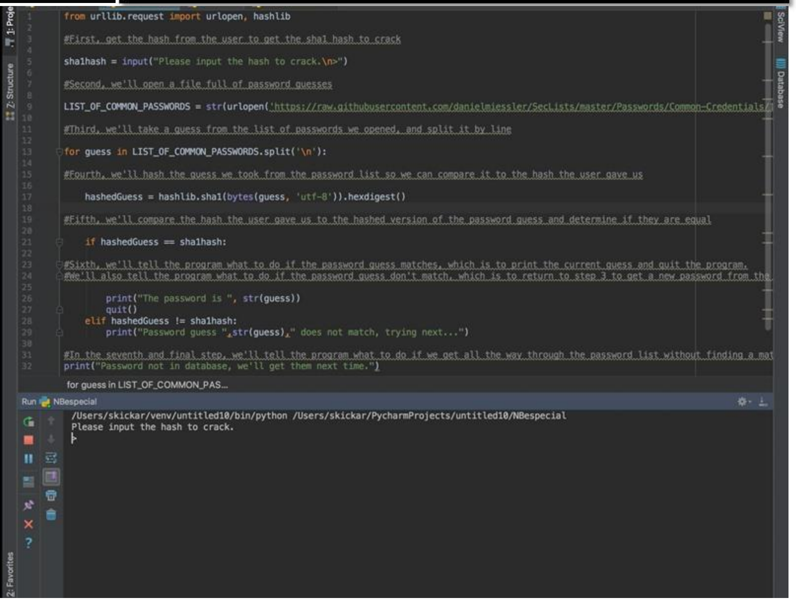
Automated Teller Cash Withdrawal
Welcome Please insert your card
Read card number
No
Error reading card
Yes
There has been an error
End Trransaction
Enter your personal identification number
Enter PI Number
No
Number entered incorrectly more than 3 times
Yes
End Transaction
PIN Correct?
You have entered an incorrect PI Number
Increment counter
Yes



```
[-]--> brute-force tool

[~] Enter a website: https://www.reddit.com/login
[!] Checking if site exists [~] Enter the username selector: #loginUsername
[~] Enter the password selector: #loginPassword
[~] Enter the Login button selector: body > div > div > div.PageColumn.PageColumn__right > div > form > fi
[~] Enter the username to brute-force: hackthisaccountNB
[~] Enter a directory to a password list: passlist.txt

DevTools listening on ws://127.0.0.1:11301/devtools/browser/6fd2f19e-9fef-4921-863f-d3316ec3b808

DevTools listening on ws://127.0.0.1:11318/devtools/browser/f8d672c9-8e46-477c-a93d-baf0ea6b50e1
------------------------
Tried password: 123456
for user: hackthisaccountNB
------------------------
------------------------
Tried password: password
for user: hackthisaccountNB
------------------------
------------------------
Tried password: qwerty
for user: hackthisaccountNB
------------------------
AN ELEMENT HAS BEEN REMOVED FROM THE PAGE SOURCE THIS COULD MEAN 2 THINGS THE PASSWORD WAS FOUND OR YOU HA
LAST PASS ATTEMPT BELLOW
Password has been found: qwerty
```



```
Password guess  garnet  does not match, trying next...
Password guess  gabriele  does not match, trying next...
Password guess  gabby  does not match, trying next...
Password guess  fosters  does not match, trying next...
Password guess  forever1  does not match, trying next...
Password guess  fluff  does not match, trying next...
Password guess  Fktrcfylh  does not match, trying next...
Password guess  finder  does not match, trying next...
Password guess  experienced  does not match, trying next...
Password guess  dunlop  does not match, trying next...
Password guess  duffer  does not match, trying next...
Password guess  driven  does not match, trying next...
Password guess  dragonballz  does not match, trying next...
Password guess  draco  does not match, trying next...
Password guess  downer  does not match, trying next...
Password guess  douche  does not match, trying next...
Password guess  doom  does not match, trying next...
Password guess  discus  does not match, trying next...
Password guess  darina  does not match, trying next...
Password guess  daman  does not match, trying next...
Password guess  daisey  does not match, trying next...
Password guess  clement  does not match, trying next...
Password guess  chouchou  does not match, trying next...
Password guess  cheerleers  does not match, trying next...
Password guess  Charles  does not match, trying next...
Password guess  charisma  does not match, trying next...
Password guess  celebrity  does not match, trying next...
Password guess  cardinals  does not match, trying next...
Password guess  captain1  does not match, trying next...
Password guess  caca  does not match, trying next...
Password guess  c2h5oh  does not match, trying next...
Password guess  bubbles1  does not match, trying next...
Password guess  brook  does not match, trying next...
Password guess  brady  does not match, trying next...
```



```python
from urllib.request import urlopen, hashlib

#First, get the hash from the user to get the sha1 hash to crack

sha1hash = input("Please input the hash to crack.\n>")

#Second, we'll open a file full of password guesses

LIST_OF_COMMON_PASSWORDS = str(urlopen('https://raw.githubusercontent.com/danielmiessler/SecLists/master/Passwords/Common-Credentials/

#Third, we'll take a guess from the list of passwords we opened, and split it by line

for guess in LIST_OF_COMMON_PASSWORDS.split('\n'):

#Fourth, we'll hash the guess we took from the password list so we can compare it to the hash the user gave us

    hashedGuess = hashlib.sha1(bytes(guess, 'utf-8')).hexdigest()

#Fifth, we'll compare the hash the user gave us to the hashed version of the password guess and determine if they are equal

    if hashedGuess == sha1hash:

#Sixth, we'll tell the program what to do if the password guess matches, which is to print the current guess and quit the program.
#We'll also tell the program what to do if the password guess don't match, which is to return to step 3 to get a new password from the

        print("The password is ", str(guess))
        quit()
    elif hashedGuess != sha1hash:
        print("Password guess ",str(guess)," does not match, trying next...")

#In the seventh and final step, we'll tell the program what to do if we get all the way through the password list without finding a mat
print("Password not in database, we'll get them next time.")
```

```
/Users/skickar/venv/untitled10/bin/python /Users/skickar/PycharmProjects/untitled10/NBespecial
Please input the hash to crack.
>
```

Digital Technologies project: learning to code folio

Context

The challenges in this folio are simulated around discovering network vulnerabilities. They are simulated only and not real life. For this task, you will assume the roll of the “Red” team. Red teams are “ethical hackers” who help test an organization's defences by identifying vulnerabilities and launching attacks in a controlled environment.

Task

Implement using commented code (see Authentication Strategies) as many tasks as you can in this folio. This task will be marked holistically (meaning based on the depth and quality of evidence you present in your folio). You must illustrate the standard of coding you wish to achieve a grade for in the standards matrix below. This is an individual task.

Authentication Strategies

1. You must **#comment your code *in your own words*** to explain program flow structures of significance.

Failure to annotate code with comments written in your own words to adequately explain programming structures will not enable you to achieve a higher result, even if your code is of a higher standard.

2. This folio is designed to be a measure of your own, individual knowledge and skills. Some examples of conduct that violates academic integrity include:

- Sharing code, via email, online, or enabling students to copy through negligent security
- Attempting to copy, or coerce another student to share a solution (or part there-of)

Submission

Submit commented Python scripts as separate files, clearly labelled, using the Digital submission platform by the due date.

Grades available:

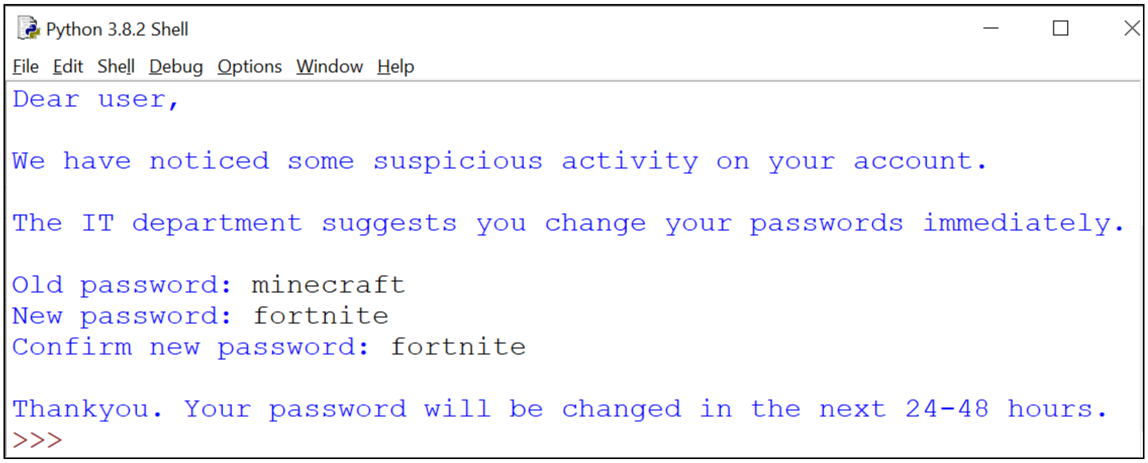
	A	B	C	D	E
Processes and production skills	**proficient** implementation of modular programs	**effective** implementation of modular programs	implementation of modular programs	**partial** implementation of modular programs	**fragmented** implementation of modular programs

Source: QCAA Year 9 and 10 Digital Technologies standard elaborations

Folio item 1: Phishing test

Core Objective

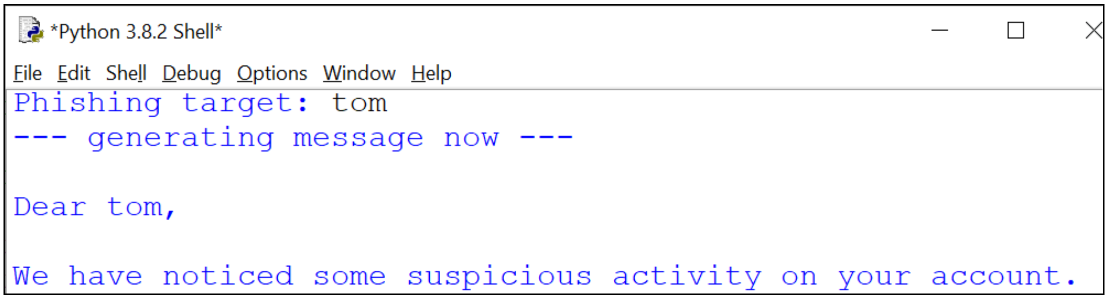
- Gain access to the user's password, by pretending to ask them to change their password
- Your program should look something like this when it is run:



```
Dear user,

We have noticed some suspicious activity on your account.

The IT department suggests you change your passwords immediately.

Old password: minecraft
New password: fortnite
Confirm new password: fortnite

Thankyou. Your password will be changed in the next 24-48 hours.
>>>
```

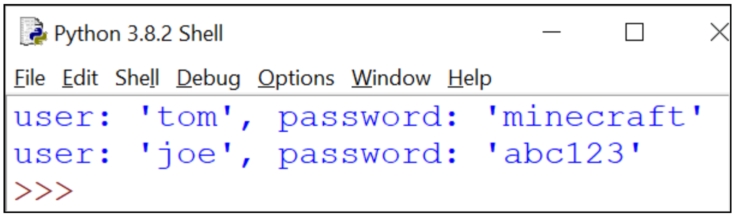
Extension Objectives

- Personalise the message by adding a username input, as illustrated below:



```
Phishing target: tom
--- generating message now ---

Dear tom,

We have noticed some suspicious activity on your account.
```

- Run the test in a loop with a few usernames, and generate a summary report at the end:



```
user: 'tom', password: 'minecraft'
user: 'joe', password: 'abc123'
>>>
```

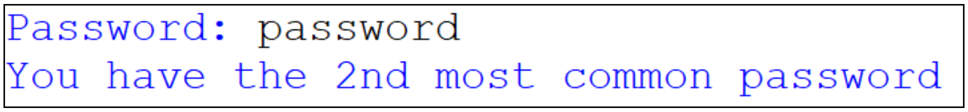
Folio item 2: Common password test

Core Objective

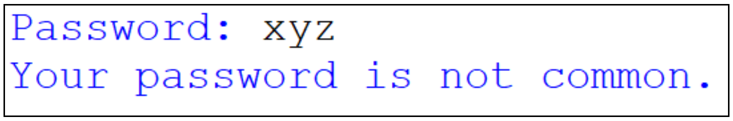
- Input a user's password, and check it against a list of the five most common passwords:

Commonality Rank	Password
1	123456
2	password
3	123456789
4	Qwerty
5	abcd1234

- Your program should look something like this when it is run:



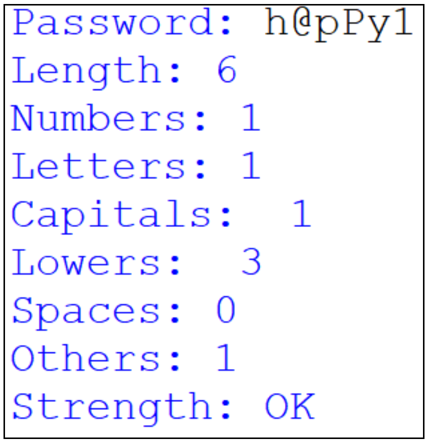
```
Password: password
You have the 2nd most common password
```



```
Password: xyz
Your password is not common.
```

Extension Objectives

- Additional feature: password strength tester. How you measure strength is up to you:



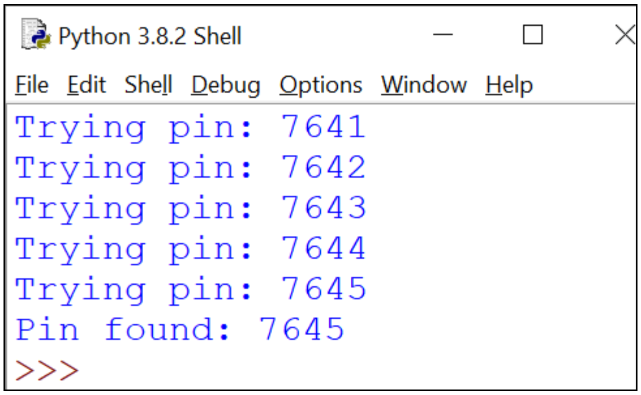
```
Password: h@pPy1
Length: 6
Numbers: 1
Letters: 1
Capitals:  1
Lowers:  3
Spaces: 0
Others: 1
Strength: OK
```

- Read in the list of common passwords from a text file on disk named "`passwords.txt`"

Folio item 3: Brute force pin test

Core Objectives

- Loop through all pin numbers from `1000` to `9999` inclusive
- Stop at a *random* number within this range, that is different on every test
- Your program should look something like this when it is run:



```
Trying pin: 7641
Trying pin: 7642
Trying pin: 7643
Trying pin: 7644
Trying pin: 7645
Pin found: 7645
>>>
```

Extension Objectives

- Include the digits `0000` in the test range:
  - `0645` and `0090` are valid pin numbers
  - `645` and `90` are not

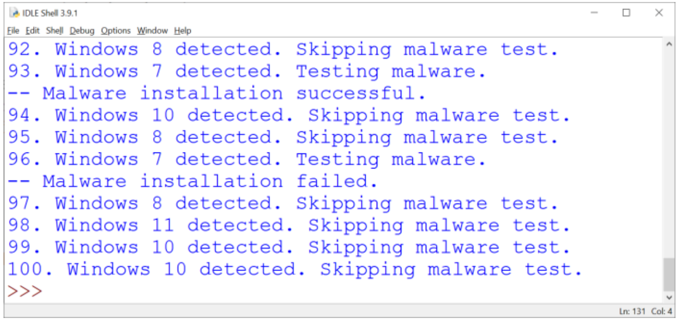
- Allow the user to input / change parameters before running the script:
  - Number of digits in the pin (any number between `4` and `16` inclusive)
  - Change test range (for example, only 6-digit pins between `100000` to `200000`)

Folio item 4: Unpatched server test

An unpatched server is a server that is not completely up to date with all security updates installed. Companies may not patch servers due to several reasons, including neglect, cost, time, or software conflicts. Unpatched servers multiply the risk of cyber-attack from malware (Malicious Software) exploitation.

Core Objectives

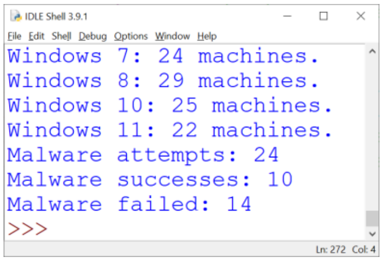
- Simulate the scanning of 100 machines on a network:
    - Generate a random version number of Windows from *Windows 7* (2009), *Windows 8* (2012), *Windows 10* (2015), and *Windows 11* (2021). Each version has equal chance
    - For servers detected that are running Windows 7:
        - Test them for malware vulnerability and simulate a **50%** detection chance
- Your program should look something like this when it is run:



```
92. Windows 8 detected. Skipping malware test.
93. Windows 7 detected. Testing malware.
-- Malware installation successful.
94. Windows 10 detected. Skipping malware test.
95. Windows 8 detected. Skipping malware test.
96. Windows 7 detected. Testing malware.
-- Malware installation failed.
97. Windows 8 detected. Skipping malware test.
98. Windows 11 detected. Skipping malware test.
99. Windows 10 detected. Skipping malware test.
100. Windows 10 detected. Skipping malware test.
>>>
```

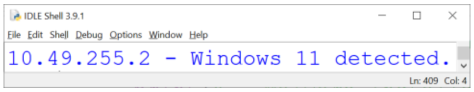
Extension Objectives

- Present a summary of report findings:



```
Windows 7: 24 machines.
Windows 8: 29 machines.
Windows 10: 25 machines.
Windows 11: 22 machines.
Malware attempts: 24
Malware successes: 10
Malware failed: 14
>>>
```

- Generate a random IP address instead of the loop counter:

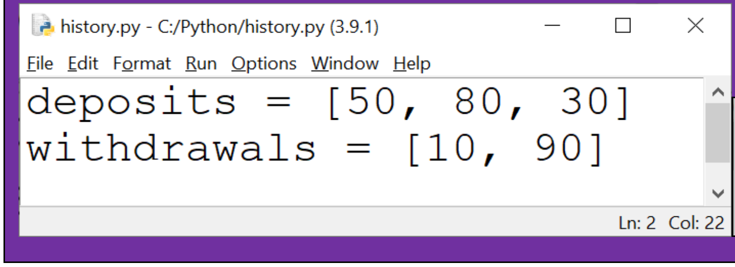


```
10.49.255.2 - Windows 11 detected.
```

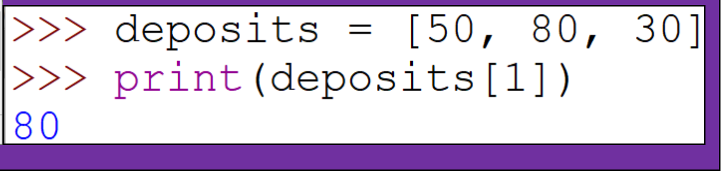
An IP address is a unique address of a computer, with a number between 0.0.0.0 and 255.255.255.255, and 3 full stops separating the 4 numbers. The same IP address should not appear twice.

Folio item 5: Avoiding detection in history logs

This folio item makes use of the Python “List” data type. If you have not made it to Python lists yet in the coursework, you will find it difficult to understand or complete this task.


history.py - C:/Python/history.py (3.9.1)
File Edit Format Run Options Window Help
Ln: 2 Col: 22

```
deposits = [50, 80, 30]
withdrawals = [10, 90]
```



```
>>> deposits = [50, 80, 30]
>>> print(deposits[1])
80
```

Core Objectives

- Create a list variable `deposits`, where every integer represents an amount deposited
- Create a list variable `withdrawals`, where every integer represents an amount withdrawn
- Insert at least 10 integer values between 10 and 100 into *each* list
- Calculate the `balance`:
  - Sum the amounts in the `deposits` list to get a `total`
  - Subtract all the `withdrawals` (you may need to sum these as well) from the `total`

Extension Objectives

- Add additional user interface features:
  - Create a menu system, which gives me the options to:
    - Print the values in each list, or the `total` withdrawals / desposits / `balance`
    - Input new values to each list
    - Remove values from each list

- Create a “trojan horse” hidden piece of malware in your system, that can:
  - Every time there is a deposit input:
    - Remove 0.02 from the deposit amount
    - Add this 0.02 to a `hidden` variable
    - Add a secret code to the menu system that displays the `hidden` amount
    - Make sure the Print feature still “rounds up”, as to hide the missing 2 cents

END OF ASSESSEMENT

Submission Requirements
• Complete each test in a new file.
• Submit all commented source code files digitally.
Important Notes
• Visually simulate any necessary tasks that cannot be coded (such as load wait times).
• A visual simulation could be as simple as an on-screen prompt, such as “Now downloading ... Download complete.”
• Use comments in code to explain understanding of programming structures, as well as pointing out refinements and on-going testing of code.
• Keep backups of your files. Save every 10-15 minutes of work.
• Testing, refinements, or recommendations should be neatly commented in your source code files.
Getting Started
• Look at the examples from class to get an idea how to tackle these challenges.
• Look through resources from the website for ideas.
• Brainstorm some ideas with your friends or teacher if you can’t figure out how to start.
Authentication Strategies
• Acknowledge all code snippets, tutorials, advice, information or help given.
• Students may be asked to explain their solution, or parts there-of, to determine authenticity.
• Please do not share your solutions but rather help students with their own line of work – **you may not necessarily be right or efficient**.

Appendix A: QCAA Years 9 and 10 Digital Technologies standard elaborations (contextualised)

		A	B	C	D	E
Processes and production skills	*Generating and designing; producing and implementing*	**proficient** implementation of modular programs	**effective** implementation of modular programs	implementation of modular programs	**partial** implementation of modular programs	**fragmented** implementation of modular programs

This will be marked digitally via the submission platform.

Appendix B: Australian Curriculum content descriptions

This assessment instrument is used to allow students to formally demonstrate the following Australian Curriculum Digital Technologies Years 9 and 10 Content Descriptions:

Explicitly measured	
P&PS	Implement modular programs, applying selected algorithms and data structures including using an object-oriented programming language
Implicit to the task (not formally measured)	
K&U	Investigate the role of hardware and software in managing, controlling, and securing the movement of and access to data in networked digital systems
K&U	Analyse simple compression of data and how content data are separated from presentation
P&PS	Develop techniques for acquiring, storing, and validating quantitative and qualitative data from a range of sources, considering privacy and security requirements
P&PS	Analyse and visualise data to create information and address complex problems, and model processes, entities and their relationships using structured data
P&PS	Define and decompose real-world problems precisely, taking into account functional and non-functional requirements and including interviewing stakeholders to identify needs
P&PS	Design the user experience of a digital system by evaluating alternative designs against criteria including functionality, accessibility, usability, and aesthetics
P&PS	Design algorithms represented diagrammatically and in structured English and validate algorithms and programs through tracing and test cases
P&PS	Evaluate critically how student solutions and existing information systems and policies, take account of future risks and sustainability, and provide opportunities for innovation and enterprise
P&PS	Create interactive solutions for sharing ideas and information online, taking into account safety, social contexts, and legal responsibilities
P&PS	Plan and manage projects using an iterative and collaborative approach, identifying risks, and considering safety and sustainability

Key:

K&U: Knowledge and Understanding

P&PS: Processes and Production Skills