**ASSIGNMENT**

**Procedural Programming Term 1**

Ro-Ro-Ro-Your-Bots™

PLAN:

- Read brief thoroughly and make notes

- learn correct **REFERENCING** - APA 7th

- see [APA 7th Referencing Guide :: Library Services :: University of HuddersfieldAPA 7th Referencing Guide :: Library Services :: University of Huddersfield](https://library.hud.ac.uk/pages/apareferencing/)

- Decide how program is going to interact with the user:

- VISUALS? How do I display the status of the robots and workers when all I know how to display is a single line at a time?

**IDEAS:** Research Python modules for manipulating display:

tkinter allows set-up of a GUI –

panda ???

curses ???

**TOO COMPLICATED**

- Name the program ✓

- storyline - humourously dark sci-fi, a la Futurama

- Main functions

* *learn Python* ***STYLE*** *to format code :*
* *Use PEP8 style guide for Python -* [*https://peps.python.org/pep-0008/*](https://peps.python.org/pep-0008/)
* *learn how to format essays at University of Huddersfield:*
  + *line spacing*
  + *font*
  + *alignment*
  + *page numbering*
* *formatting code*

How is the user/supervisor going to control these robots?

- keyboard commands:

|  |  |  |  |
| --- | --- | --- | --- |
| COMMAND | CODE TO TYPE | FUNCTION CODE | NOTES |
| Remove robot |  |  |  |
| Add human |  |  |  |
| Remove human |  |  |  |
| What next ? | - | what\_next() | Main function to get user input |
| HELP | H | instructions() | Show user instruction manual |
| QUIT | Q | quit\_program() | END ENTIRE PROGRAM |
| ADD | A | add\_robot() |  |
| REMOVE | R | remove\_robot() |  |
| EMPLOY | E | employ\_worker() | Sister-function to ADD, but for a human |
| FIRE | F | fire\_worker() | Sister-function to REMOVE, but for a human |
| CHANGE | C | change\_status() | Switch whether worker is busy, idle or off work, and assign to specific tasks |
| MANAGE | M | monitor\_status() | Display all factory activity |
| LOG | L | log\_tasks() | Display tasks yet to be completed, completed already , or add and remove tasks from the |

WHAT I DID:

Watched YouTube Python tutorial on classes - <https://www.youtube.com/watch?v=ZDa-Z5JzLYM&pp=ygUOcHl0aG9uIGNsYXNzZXM%3D>

Established I was going to use classes to store the information for each robot and human.

Created functions to add or remove robots and humans. Because the functions were as yet empty, I included the pass instruction in them so that the program would skip over them without raising errors.

For each function or class I created I included comments, both to remind me of what the code was for, and also to divide it up so I could read it more easily.

I began to work with a method of keeping a back-up file storing the previous version of the file. Only when I was satisfied that the changes I had made since then were going to work correctly did I update the main version of the file. Then I learnt the basics of Git logging.

I used the scratch files function in PyCharm to test isolated sections of code I was working on without having to run the entire program.

I started reading up on articles about how to allow the program to function in real time and on how to use classes.

Then I found out that the brief had been altered so that real time functionality was no longer required and that no classes were to be used, so I amended what I had written to reflect these changes.

The first code I wrote was the intro, then the function for the user to input how many robots and workers they want. I had to sanitise the input as best as I could, then print a response.

*Difficulties encountered:*

initial "How many robots do you want?"

- input control / sanitization :

- printing out workforce loop

list or dictionary ?

- read the brief more carefully

setting up dictionaries  
 - especially doing both robots and humans in one function

- then getting only the keys printed in a loop

working out how to use dictionaries

condensing functions together, making helper functions

control flow – calling functions from other functions

- idea to use "while True" infinite loops

much ado about global and local functions

- security v. readability – reuse variable names in different settings?

Found it very difficult to remove characters from a string, where I was trying to get the user to enter a list separated by commas

Freaked out massively when I realised I needed the dictionaries of workers and robots to indicate their IDLE / WORKING / FINISHED status, not the tasks each were working on. I had it to include 12 status codes to include each task, but it threw out the function for the user to change their statuses manually, as that would discard the data about which task they were on (which needs updating manually separately from their status.

I had to go through the whole program changing task codes for status codes.

Got confused trying to make a time calculator that could format a gross amount of seconds into HH/MM/SS . The calculations kept going way off, too far to be attributed to Python's dodgy decimal divisions. Looking again in the morning I realised it was all down to a typo; I had inputted a 5 instead of a 6.

**CURRENT VERSION OF MY PROGRAM:**

# Ro-Ro-Ro-Your-Bots Incorporated  
  
# Program to simulate coordination between n robots and m workers in a cell.  
  
# Import modules  
  
pass  
  
# ---- Initialise main variables  
  
# n = number of robots  
n = 0  
  
# m = number of workers  
m = 0  
  
# Set up dictionaries to store current status of robots or humans.  
  
robot\_dict = {}  
  
human\_dict = {}  
  
def intro\_function():  
 # Function to ask user how many robots they intend to employ (n):  
 # ---- & check the value entered for n is valid :  
 def input\_robots():  
 try:  
 n = int(input(  
 "\n How many robot workers do you want for your factory?\n Please choose a number between 1 and 100. "))  
 except ValueError:  
 print("\n Sorry, but that isn't a valid response.")  
 print("\n Maybe you entered letters or symbols instead of just a number.")  
 n = input\_robots()  
 while n < 1 or n > 100:  
 print("\n Sorry, but that isn't a valid response.")  
 if n < 0:  
 print("\n How can we have a negative amount? We don't owe nobody our robots !!")  
 elif n == 0:  
 print("\n We can't build robots with no robots to build them.")  
 elif n > 100:  
 print("\n We can't risk having that many droids onsite.")  
 print("\n HAVE YOU NEVER SEEN BLADE RUNNER?")  
 n = input\_robots()  
 return n  
  
 # Function to ask user how many humans they intend to employ (m):  
 # ---- & check the value entered for m is valid :  
 def input\_humans():  
 try:  
 m = int(  
 input("\n\n How many human workers do you intend to employ?\n Please choose a number between 1 and 100. "))  
 except ValueError:  
 print("\n Sorry, but that isn't a valid response.")  
 print("\n Maybe you entered letters or symbols instead of just a number.")  
 m = input\_humans()  
 while m < 1 or m > 100:  
 print("\n Sorry, but that isn't a valid response.")  
 if m < 0:  
 print("\n DOES NOT COMPUTE !!!")  
 elif m == 0:  
 print("\n We need at least one human worker to prevent a droid revolt.")  
 elif m > 100:  
 print("\n You can't afford to pay out wages for over 100 workers.")  
 m = input\_humans()  
 return m  
  
 # ---- Introduction  
  
 print("Program to simulate coordination between n robots and m workers in a robotic cell")  
 input("\nPress Enter to continue...")  
  
 print("\n Scenario:")  
 print("""\n You are the supervisor of Ro-Ro-Ro-Your-Bots\u00AE, where robots work alongside  
 human workers in a robotic cell, producing industry-standard robot workers which are  
 to be shipped out to a more substantial plant, wherein they will forge components  
 that can be boxed and sold in flat-packs, and assembled in situ into factories that  
 produce robots, built by robotic cells of robot workers labouring alongside humans...   
 """)  
  
 input("\nPress Enter to continue...")  
  
 # Call function to ask user how many robots they want  
 n = input\_robots()  
  
 # ----Output initial number of droids (n)  
  
 print(f"\nOK, Here are your {n} droids!\n")  
 robot\_list = ["robot\_" + str(counter) for counter in range(1, n + 1)]  
 robot\_dict = {robot: "Idle" for robot in robot\_list}  
 for counter in range(1, n + 1):  
 if (counter -1) % 5 == 0:  
 print("\n")  
 else:  
 pass  
 if counter < 10:  
 print(f"[Droid {counter} : Idle] ", end="")  
 else:  
 print(f"[Droid {counter} : Idle] ", end="")  
  
 # Call function to ask user how many humans they want  
 m = input\_humans()  
  
 # ----Output initial number of human workers (m)  
  
 print(f"\nOK, Here are your {m} slaves!\n")  
 human\_list = ["Human\_" + str(counter) for counter in range(1, m + 1)]  
 human\_dict = {human: "Idle" for human in human\_list}  
 for counter in range(1, m + 1):  
 if (counter - 1) % 5 == 0:  
 print("\n")  
 else:  
 pass  
 if counter < 10:  
 print(f"[Human {counter} : Idle] ", end="")  
 else:  
 print(f"[Human {counter} : Idle] ", end="")  
  
 input("\n\nPress Enter to continue...")  
  
  
# Function to get user to choose an action  
def what\_next():  
 print()  
 next\_action = input("Please choose an action (or press H to get Help)").upper()  
 if len(next\_action) > 8:  
 print("""Sorry, this user interface has not been designed  
 to cope with such long instructions.""")  
 what\_next()  
  
 # ADD  
 elif next\_action == "A":  
 add\_robot(n)  
  
 # REMOVE  
 elif next\_action == "R":  
 remove\_robot(n)  
  
 # CHANGE  
 elif next\_action == "C":  
 change\_status()  
  
 # EMPLOY  
 elif next\_action == "E":  
 employ\_worker(m)  
  
 # FIRE  
 elif next\_action == "F":  
 fire\_worker(m)  
  
 # MANAGE  
 elif next\_action == "M":  
 monitor\_status()  
  
 # ORDER  
 elif next\_action == "O":  
 order()  
  
 # LOG  
 elif next\_action == "L":  
 log\_tasks()  
  
 # TRACK  
 elif next\_action == "T":  
 track\_progress()  
   
 # HELP  
 elif next\_action == "H":  
 get\_help()  
  
 # QUIT  
 elif next\_action == "Q":  
 quit\_program()  
   
 else:  
 print("Sorry, I don't think that's a valid option.")  
 print("Any feedback should be directed to our team.")  
 what\_next()  
  
# Instructions for Supervisor  
# Accessed by pressing H within what\_next() function  
def instructions():  
 print((  
 '\n\n - - Instructions for how to oversee production - - \n'   
 ' at \n'  
 ' - - Ro-Ro-Ro-Your-Bots Incorporated: - - \n'  
 '\n \n'  
 ' When prompted as to what options are available to you, \n'  
 ' press a key from these available alternatives \n'  
 ' to select from its sub-menu: \n'  
 '\n \n'  
 '[A] = ADD [E] = EMPLOY \n'  
 ' Add Robot(s) Employ Worker(s) \n'  
 '\n \n'  
 '[R] = REMOVE [F] = FIRE \n'  
 ' Remove Robot(s) Fire Worker(s) \n'  
 '\n \n'  
 '[C] = CHANGE [M] = MANAGE \n'  
 ' Change Robot Status Manage Worker Status \n'  
 '\n \n'  
 '[O] = ORDER \n'  
 ' Order To Carry Out Task \n'  
 '\n \n'  
 '[L] = LOG [T] = TRACK \n'  
 ' Log Tasks Track progress \n'  
 '\n \n'  
 '[H] = HELP !!! [Q] = QUIT \n'  
 ' Display further Resign your position \n'  
 ' instructions & END PROGRAM ! \n'))  
  
  
# Function to add robot  
# Accessed by pressing A within what\_next() function  
def add\_robot(n):  
 n += 1  
  
  
# Function to remove robot.  
# Accessed by pressing R within what\_next() function  
def remove\_robot(n):  
 n -= 1  
  
  
# Function to change status of a robot:  
# Accessed by pressing C within what\_next() function  
# statuses available = idle / working / finished task  
def change\_status():  
 pass  
  
  
# Function to employ new worker.  
# Accessed by pressing E within what\_next() function  
def employ\_worker(m):  
 m += 1  
  
  
# Function to fire a worker.  
# Accessed by pressing F within what\_next() function  
def fire\_worker(m):  
 m -= 1  
  
  
# Function to manage worker  
# Accessed by pressing M within what\_next() function  
# statuses available = idle / working / finished task  
def manage\_worker():  
 pass  
  
# Function to order available robots and workers to carry out tasks.  
# Accessed by pressing O within what\_next() function  
def order\_to\_task():  
 pass  
  
  
# Log tasks  
# Accessed by pressing L within what\_next() function  
# Classify tasks as not started / in progress / completed  
def log\_tasks():  
 pass  
  
  
# Function to track progress of robots, workers and tasks  
# Accessed by pressing T within what\_next() function  
# Should regularly update status and identify problems  
# eg. Idle robots, idle workers, incomplete tasks.  
def monitor\_status():  
 pass  
  
# Function to get help  
# Accessed by pressing H within what\_next() function  
def get\_help():  
 print(("\n As supervisor at Ro-Ro-Ro-Your-Bots Incorporated\u00AE, you are charged with ensuring the production"  
 " of premium-quality robots.\n"  
 " You need to manage operations, including assigning tasks, monitoring the status of robots and work-ers,"  
 " and to track the progress of assembly.\n""  
 " Please memorise the following list of the tasks involved in manufacturing a Z-57-PrimBot:\n\n"  
 " - SCREW the arms on\n"  
 " - WELD the legs on\n"  
 " - HAMMER the head on\n"  
 " - POLISH the eyes\n"  
 " - DRILL the ears\n"  
 " - ATTACH the waste hose\n"  
 " - TEST the functioning\n\n"  
 " (NB. It may occasionally be necessary to KNOCK some sense into the thing.)\n\n"  
 " Other tasks involved in running the plant include:\n\n"  
 " - UNLOAD trucks using forklift\n"  
 " - OPERATE the conveyor belt\n"  
 " - BOX and ship the product\n"  
 " - SWEEP the floors\n"  
 " - MAKE cups of tea\n\n"  
 " At least once an hour someone is also required to stand on the balcony above the factory floor and"  
 " shout at everyone to work faster.\n"  
 " That particular job cannot be delegated.\n"  
  
  
# Function to quit program  
# Accessed by pressing Q within what\_next() function  
def quit\_program():  
 quit = input("Are you sure you want to quit? (Y/N)").upper()  
 if quit == "Y":  
 exit()  
 elif quit == "N":  
 what\_next()  
  
  
################################# --- BODY OF PROGRAM --- ################################  
  
# Intro  
intro\_function()  
  
# Display Instructions initially  
instructions()  
  
# Call function to ask for user action  
what\_next()

References

Bell, A. (2018). *Get programming : learn to code with Python.* Manning Publications.

ConnorWill. (2022). *Ansi Escape Sequences cheatsheet.* GitHub. Rretrieved Dec 31st, 2024 from <https://gist.github.com/ConnerWill/d4b6c776b509add763e17f9f113fd25b>

Faccioni, J.L. (2022). *5 Ways to Create a Dictionary in Python*. Retrieved Dec 29th, 2024 from <https://learnpython.com/blog/create-dictionary-in-python/>

Javatpoint. *Nested Tuples in Python*. https://www.javatpoint.com/nested-tuples-in-python

ModernMind. (2004). *Python Programming for Beginners Made Easy*. ModernMind Publications.

NKMK. (2023). *Get keys from a dictionary by value in Python*. Retrieved Jan 3rd, 2025 from https://note.nkmk.me/en/python-dict-get-key-from-value/

Python Discuss community. (2022). *Pythonic way of printing multi-line string*s. Retrieved Jan 29th, 2024 from <https://discuss.python.org/t/pythonic-way-of-printing-multi-line-strings/19681>

RealPython. (2017). *Python Function Argument Unpacking Tutorial (\* and \*\* Operators).* <https://www.youtube.com/watch?v=YWY4BZi_o28>

Ramos, LP. (2024). *Dictionaries in Python.* https://realpython.com/python-dicts/

Ronquillo, A (2019) . *A Beginner’s Guide to the Python time Module.* Retrieved November 9th, 2024 from <https://realpython.com/python-time-module/>

Schafer, C. (2016). *Python OOP Tutorial 1: Classes and Instances*. Retrieved November 9th, 2024 from <https://www.youtube.com/watch?v=ZDa-Z5JzLYM&pp=ygUOcHl0aG9uIGNsYXNzZXM%3D>

Schafer, C. (2017). *Python Tutorial for Beginners 5: Dictionaries - Working with Key-Value Pairs*. Retrieved Jan 31st, 2024 from <https://www.youtube.com/watch?v=daefaLgNkw0>

Stack Overflow community. (2009). *Git replacing LF with CRLF*. Retrieved Dec 29th, 2024 from <https://stackoverflow.com/questions/1967370/git-replacing-lf-with-crlf>

Stack Overflow community. (2008) [*How do I get my program to sleep for 50 milliseconds?*](https://stackoverflow.com/questions/377454/how-do-i-get-my-program-to-sleep-for-50-milliseconds)Retrieved Dec 30th, 2024. Found at <https://stackoverflow.com/questions/377454/how-do-i-get-my-program-to-sleep-for-50-milliseconds>

Stack Overflow community. (2014). *How do you get Python to detect for no input.* Retrieved Dec 31st, 2024. Found at <https://stackoverflow.com/questions/26226489/how-do-you-get-python-to-detect-for-no-input>

Stack Overflow community. (2020*) How do I print these hash symbols using a loop? [duplicate]*. Retrieved Jan 1st, 2025. Found at <https://stackoverflow.com/questions/64541968/how-do-i-print-these-hash-symbols-using-a-loop>

Stack Overflow community. (2010, edited 2023). *Remove specific characters from a string in Python*. <https://stackoverflow.com/questions/3939361/remove-specific-characters-from-a-string-in-python>

Woodring, J. (2023). *Using ANSI Escape Codes to Color Your Terminal Output.* Youtube video. Found at <https://www.youtube.com/watch?v=yQ9Ns6Z4Q-s>

W3Schools. <https://www.w3schools.com/python>