Summer Challenge

# Python Cheat sheet

|  |  |
| --- | --- |
| Python skill | How does it look |
| Input |  |
| Output |  |
| If |  |
| While |  |
| For |  |
| Function |  |
| Reading from a file |  |
| Writing to a file |  |

# Simple tasks

## Grade scores

Write a Python program that asks the user for a score and prints a grade based on the following:

* 80–100 A
* 60–79 B
* 40–59 C
* 30–39 D
* < 30 U

## Vowels

calculate the vowel value of a word based on the following rubric:

* a 5 points
* e 4 points
* i 3 points
* o 2 points
* u 1 points

## Long Name

Define a function called “long\_name()” that takes a name input and returns a Boolean value (true or false) based on the number of characters in the name. Assume a name is long if it contains more than 14 characters

## Larger

Define a function called “larger()” that takes two integers and returns the larger of the two.

Can you extend the program so that it

## Binary Task

Write a program that will input a binary number and output a denary value

### Challenge

Write a program that will accept two binary numbers add the binary numbers and output the answer in binary

## Vending machine Task

A food vending machine accepts 10p, 20p, 50p and £1 coins. One or more coins are inserted and the current credit is calculated and displayed. A product is selected from those available. The system checks to see if there is enough credit to Purchase the product chosen. If there is not enough credit the system displays an error message. If there is enough credit it dispenses the product, updates the credit available and displays the remaining credit. Further selections can be made if there is enough credit. The vending machine simulation should have five products and prices

### Challenge

The vending machine products should be stored in a file and then with a stock value. When people buy a product this should be take away from the stock value and file should be updated.

## Password Task

1. Create a system to accept and test a password for certain characteristics.
   * It should be at least 6, and no more than 12 characters long.
   * A message to indicate that the password is acceptable must be displayed.
   * The system must indicate that the password has failed and why, asking the user to re enter their choice until a successful password is entered.
   * The user must be able to change passwords and add new passwords if they are longer than 12 characters or less than 6 characters than this should be rejected.
2. Password strength can be assessed against simple criteria to assess its suitability; for example a password system using only upper and lower case alphabetical characters and numeric characters could assess the password strength as:
   * WEAK if only one type used, e.g. all lower case or all numeric
   * MEDIUM if two types are used
   * STRONG if all three types are used.
   * For example:
     + hilltop, 123471324, HAHGFD are all WEAK
     + catman3 and 123456t are MEDIUM
     + RTH34gd is STRONG
3. A message to indicate the password strength should be displayed after an acceptable password is chosen.
4. Write the password to a file
5. Extend the program so that if its already been used it cannot be used again.