Back-to-Basics Revision: Dictionaries

Task 1

The task is to create a dictionary and check the membership of a given item in the dictionary.

Task 1.1

Write a function `make_dict` that:

- takes a list as a parameter
- stores the items of the list as keys in a dictionary
- assigns a random integer value to each key.

Task 1.2

Create a dictionary `test_dict` using the following list as an argument:

```
['brave', 'calm', 'delightful', 'eager', 'faithful', 'gentle', 'happy',
'obedient', 'thankful', 'victorious', 'witty', 'zealous'].
```

Task 1.3

Write additional code to display an indication whether the following items are in `test_dict`:

- `'brave'`
- `'palm'`
- `'happy'`
- `'fun'`

Task 2

The task is to extract items common to two dictionaries.

Task 2.1

Write a function `dict_intersect` that:

- takes two dictionaries as arguments
- returns a dictionary containing only the key-value pairs found in both of the original dictionaries.

Test you function using the following dictionaries:

- `{'red':4, 'green':4, 'blue':5, 'yellow':2, 'orange':5}`
- `{'red':3, 'green':4, 'blue':4, 'yellow':2, 'orange':6}`

Task 3

The task is to determine the quantity of unique values stored in a dictionary.

Task 3.1

Write a function `count_values` that:

- takes a single dictionary as an argument
- returns the number of distinct values it contains.

Test your function with the following dictionary:

```
{'red':4, 'green':4, 'blue':5, 'yellow':2, 'orange':5},
```

The expected return value is `3`.

Task 4

The task is to invert a dictionary.

Task 4.1

Write a function `invert_dict` that:

- inverts the key-values pair in a dictionary
- returns the inverted dictionary.

Your function should account for repeated values e.g. the dictionary

```
        {'M':1, 'I':2, 'S': 3, 'P':1, 'E':1, 'L':2, 'N':1, 'G':1}
```

when inverted becomes

```
          {1:['M', 'P', 'E', 'N', 'G'], 2:['I', 'L'], 3:['S']}
```

Test your function with the aforementioned example.

Task 5

The task is to collect data on dice roll simulation.

Task 5.1

Write a function `roll`:

- that simulates the rolling of a pair of six-sided dice
- has no parameters
- returns the total of the two dice rolls.

Task 5.2

Write a main program that uses the function `roll` to simulate the rolling of two six-sided dice 1000 times.

Write additional code to:

- count the number of times that each total occurs.
- display a table that summarises this data
- express the frequency for each total as a percentage of the total number of rolls
- display the probability of obtaining each total as a percentage.

A sample output is as follows:

```
Total        Simulated Percent        Expected Percent
2            2.90                     2.78
3            6.90                     5.56
4            9.40                     8.33
...          ...                      ...
```