Pour chacune des fonctions écrites ci-dessous, on pourra utiliser `Python Tutor` pour une meilleure visualisation.

Exercice 1

1. L' instruction qui permet d'obtenir le nombre d'animaux de Gaston est : `ferme_gaston['cheval']`
2. Pour ajouter 12 poules à `ferme_Gaston` on saisit : `ferme_gaston['poule'] = 12`
3. Pour obtenir la liste des animaux de la ferme de Gaston on saisit : `list(ferme_gaston.keys()).`
4. La fonction suivante convient :

```
def nombre_animaux(ferme) :
    nb = 0
    for animal in ferme :
      nb += ferme[animal]
    return nb
```

5. La fonction suivante convient :

```
def nombre_animal(ferme,animal) :
    if animal in ferme :
      return ferme[animal]
    return 0
```

Exercice 2

1. Le dictionnaire `BDPrix` est donné par :

```
BDPrix = {'Sabre laser': 229.0,'Etoile de ninja': 29.95,
          'Cape': 75.0,'Baguette': 35.0,'Chapeau': 12.0,
          'Bandeau': 5.5,'Balai': 130.0}
```

2. La fonction suivante convient :

```
def Dispo(p, D):
  '''
  Renvoie vrai si la le produit p (clé) est dans le dictionnaire D, faux sinon.
  Le dictionnaire D est un dictionnaire de produits (str) / prix (float)
  '''
  return p in D.keys()
```

3. La fonction suivante convient :

```
def PrixMoyen(D):
  '''
  Calcule le prix moyen des produits (clés) du dictionnaire D.
  Les valeurs associées aux clés sont les prix des produits
  '''
  return sum(D.values())/len(D)
```

4. La fonction suivante convient :

```
def IntervallePrix(D, m, M):
  '''
  Renvoie la liste des produits (clés) dont le prix (valeur) est compris
  entre m et M (int ou float). D est le dictionnaire de produits/prix.
  '''
  ListeProduits = []
  for cle in D.keys()
    if m <= D[cle] <= M:
      ListeProduits.append(cle)
  return ListeProduits
```

5. Le panier correspondant à l'achat de deux sabres lasers, de trois étoiles de ninja et d'un bandeau est :

```
Panier = {'Sabre laser': 2,'Etoile de ninja': 3,'Bandeau': 1}
```

6. La fonction suivante convient :

```
def TousDispo(D1, D2):
    '''
    Renvoie True si tous les produits du panier D2 sont en vente dans le
    magasin dont le dictionnaire produits/prix est D1, False sinon.
    '''
    for cle in D2.keys():
        if cle not in D1.keys():    # Si une clé de D2 n'est pas dans D1
            return False    # alors au moins un produit n'est pas disponible
    return True
```

7. La fonction suivante convient :

```
def Total(D1, D2) :
    '''
    Calcule le total à payer à partir des produits du panier D2 et la
    quantité souhaitée ainsi que les prix de chacun des produits.
    D1 est le dictionnaire de produits/prix (celui du magasin en ligne)
    D2 est le dictionnaire de produits/quantité (le panier)
    '''
    if TousDispo(D1, D2):    # calcul uniquement si tous les produits sont disponibles
        S = 0    # initialise la somme à payer à 0 (accumulateur)
        for cle in D2.keys():
            S += D2[cle] * D1[cle]    # ajoute le prix x la quantité
        return S
    else:
        return 'Erreur : tous les articles ne sont pas disponibles'
```

Exercice 3

1. Le dictionnaire `MesRecettes` est donné par :

```
MesRecettes = {'Gâteau au chocolat':['chocolat','oeuf','farine','sucre','beurre'],
               'Gâteau au yaourt':['yaourt','oeuf','farine','sucre'],
               'Crêpes':['oeuf','farine','lait','bière'],
               'Quatre-quarts':['oeuf','farine','beurre','sucre'],
               'Kouign amann':['farine','beurre','sucre']}
```

2. La fonction suivante convient :

```
def NBIngredients(D, nom):
    '''
    Renvoie le nombre d'ingrédients d'une recette nom du dictionnaire D
    '''
    return len(D[nom])
```

3. La fonction suivante convient :

```
def RechercheRecettes(D, I):
    '''
    Renvoie la liste des recettes (clés) du dictionnaire D contenant l'ingrédient I
    '''
    RecettesTrouvees = []
    for recette in D.keys():
        if I in D[recette]:
            RecettesTrouvees.append(recette)
    return RecettesTrouvees
```

4. La fonction suivante convient :

```
def RechercheRecettes2(D, I1, I2):
  '''
  Renvoie la liste des recettes (clés) du dictionnaire D contenant
  les ingrédients I1 et I2
  '''
  RecettesTrouvees = []
  for recette in D.keys():
    if (I1 in D[recette]) and (I2 in D[recette]):
      RecettesTrouvees.append(recette)
  return RecettesTrouvees
```

5. La fonction suivante convient :

```
def RechercheRecettesMulti(D, LI):
  '''
  Renvoie la liste des recettes (clés) du dictionnaire D contenant
  tous les ingrédients de la liste LI
  '''
  RecettesTrouvees = list(D.keys())
  for recette in D.keys():
    for ingredient in LI:
      if ingredient not in D[recette]:
        RecettesTrouvees.remove(recette)
  return RecettesTrouvees
```

6. La fonction suivante convient :

```
def RechercheRecettesSansMulti(D,LI):
  '''
  Renvoie la liste des recettes (clés) du dictionnaire D ne contenant
  aucun des ingrédients de la liste LI
  '''
  RecettesTrouvees = list(D.keys())
  for recette in D.keys():
    for ingredient in LI:
      if ingredient in D[recette]:
        RecettesTrouvees.remove(recette)
  return RecettesTrouvees
```

Exercice 4

1. (a) Son nom est `Salamèche`. Il faut saisir l'instruction : `mystere['Nom']`
   (b) Il a un seul type. Il faut saisir l'instruction : `mystere['Type'][0]`
   (c) Il faut saisir l'instruction : `mystere['Vitesse'] = 65`
   (d) Il faut saisir l'instruction : `mystere['Attaque'] = 52`
2. Le pokémon `fragilady` est représenté par le dictionnaire suivant :

```
fragilady = {'Nom': 'Fragilady','HP': 70,'Attaque': 60,
             'Défense': 75,'Vitesse': 90, 'Type': ['Plante']}
```

3. (a) La fonction suivante convient :

```
def attaque(pokemon):
  '''
  Renvoie la valeur d'attaque du pokemon (dictionnaire) passé en paramètre
  '''
  return pokemon['Attaque']
```

   (b) La fonction suivante convient :

```
def NbTypes(pokemon):
    '''
    Renvoie le nombre de types du pokemon (dictionnaire) passé en paramètre
    '''
    return len(pokemon['Type'])
```

(c) La fonction suivante convient :

```
def defenseSup(pokemon,val):
    '''
    Renvoie True si le pokemeon (dictionnaire) passé en paramètre a
    une valeur de défense supérieure à val, False sinon
    '''
    return pokemon['Défense'] > val
```