Etape 1 - Importation des librairies et chargement des fichiers

1.1 - Importation des librairies

```
import pandas as pd
```

1.2 - Chargement des fichiers Excel

```
population= pd.read_csv('population.csv')
dispo_alimentaire= pd.read_csv('dispo_alimentaire.csv')
aide_alimentaire= pd.read_csv('aide_alimentaire.csv')
sous_nutrition= pd.read_csv('sous_nutrition.csv')
```

Etape 2 - Analyse exploratoire des fichiers

2.1 - Analyse exploratoire du fichier population

```
print("Le tableau comporte {} observation(s) ou article(s)".format(population.shape[0]))
print("Le tableau comporte {} colonne(s)".format(population.shape[1]))
```

```
Le tableau comporte 1416 observation(s) ou article(s)
Le tableau comporte 3 colonne(s)
```

```
print("dimension de data set:", population.shape)
```

```
dimension de data set: (1416, 3)
```

```
print("nombre de colonne=", population.shape[1])
```

```
nombre de colonne= 3
```

```
population.dtypes
```

```
Zone       object
Année       int64
Valeur    float64
dtype: object
```

```
population.count()
```

```
Zone      1416
Année     1416
Valeur    1416
dtype: int64
```

```
population.head(5)
```

	Zone	Année	Valeur
0	Afghanistan	2013	32269.589
1	Afghanistan	2014	33370.794
2	Afghanistan	2015	34413.603
3	Afghanistan	2016	35383.032
4	Afghanistan	2017	36296.113

```
population['Valeur']= population['Valeur']*1000

population =population.rename(columns={'Valeur': 'Population'})
population.head(5)
```

	Zone	Année	Population
0	Afghanistan	2013	32269589.0
1	Afghanistan	2014	33370794.0
2	Afghanistan	2015	34413603.0
3	Afghanistan	2016	35383032.0
4	Afghanistan	2017	36296113.0

2.2 - Analyse exploratoire du fichier disponibilité alimentaire

```
print("Le tableau comporte {} observation(s) ou article(s)".format(dispo_alimentaire.shape[0]))
print("Le tableau comporte {} colonne(s)".format(dispo_alimentaire.shape[1]))
```

```
Le tableau comporte 15605 observation(s) ou article(s)
Le tableau comporte 18 colonne(s)
```

```
print("le nombre de colonne =", dispo_alimentaire.shape[1])
```

```
le nombre de colonne = 18
```

```
dispo_alimentaire.head(5)
```

	Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Disponibilité alimentaire en quantité (kg/personne/an)	Disponibilité de matière grasse en quantité (g/personne/jour)	Disponibilité de protéines en quantité (g/personne/jour)	Disponibilité intérieure	Exportations - Quantité	Importations - Quantité	Nourriture	Pertes	Production	Semences	Traitement	Variation de stock
0	Afghanistan	Abats Comestible	animale	NaN	NaN	5.0	1.72	0.20	0.77	53.0	NaN	NaN	53.0	NaN	53.0	NaN	NaN	NaN
1	Afghanistan	Agrumes, Autres	vegetale	NaN	NaN	1.0	1.29	0.01	0.02	41.0	2.0	40.0	39.0	2.0	3.0	NaN	NaN	NaN
2	Afghanistan	Aliments pour enfants	vegetale	NaN	NaN	1.0	0.06	0.01	0.03	2.0	NaN	2.0	2.0	NaN	NaN	NaN	NaN	NaN
3	Afghanistan	Ananas	vegetale	NaN	NaN	0.0	0.00	NaN	NaN	0.0	NaN	0.0	0.0	NaN	NaN	NaN	NaN	NaN
4	Afghanistan	Bananes	vegetale	NaN	NaN	4.0	2.70	0.02	0.05	82.0	NaN	82.0	82.0	NaN	NaN	NaN	NaN	NaN

```
dispo_alimentaire= dispo_alimentaire.fillna(0)

ligne_en_milliers_de_tonnes= ['Autres Utilisations', 'Disponibilité intérieure',   'Exportations - Quantité',  'Importations - Quantité',  'Nourriture',   'Pertes',   'Production',   'Semences',    'Traitement',   'Variation de stock']
dispo_alimentaire.loc[:,ligne_en_milliers_de_tonnes]*=1000

dispo_alimentaire.head(5)
```

	Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Disponibilité alimentaire en quantité (kg/personne/an)	Disponibilité de matière grasse en quantité (g/personne/jour)	Disponibilité de protéines en quantité (g/personne/jour)	Disponibilité intérieure	Exportations - Quantité	Importations - Quantité	Nourriture	Pertes	Production	Semences	Traitement	Variation de stock
0	Afghanistan	Abats Comestible	animale	0.0	0.0	5.0	1.72	0.20	0.77	53000.0	0.0	0.0	53000.0	0.0	53000.0	0.0	0.0	0.0
1	Afghanistan	Agrumes, Autres	vegetale	0.0	0.0	1.0	1.29	0.01	0.02	41000.0	2000.0	40000.0	39000.0	2000.0	3000.0	0.0	0.0	0.0
2	Afghanistan	Aliments pour enfants	vegetale	0.0	0.0	1.0	0.06	0.01	0.03	2000.0	0.0	2000.0	2000.0	0.0	0.0	0.0	0.0	0.0
3	Afghanistan	Ananas	vegetale	0.0	0.0	0.0	0.00	0.00	0.00	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	Afghanistan	Bananes	vegetale	0.0	0.0	4.0	2.70	0.02	0.05	82000.0	0.0	82000.0	82000.0	0.0	0.0	0.0	0.0	0.0

2.3 - Analyse exploratoire du fichier aide alimentaire

```
print("Le tableau comporte {} observation(s) ou article(s)".format(aide_alimentaire.shape[0]))
print("Le tableau comporte {} colonne(s)".format(aide_alimentaire.shape[1]))
```

```
Le tableau comporte 1475 observation(s) ou article(s)
Le tableau comporte 4 colonne(s)
```

```
aide_alimentaire.shape[1]
```

```
4
```

```
aide_alimentaire.head(5)
```

	Pays bénéficiaire	Année	Produit	Valeur
0	Afghanistan	2013	Autres non-céréales	682
1	Afghanistan	2014	Autres non-céréales	335
2	Afghanistan	2013	Blé et Farin	39224
3	Afghanistan	2014	Blé et Farin	15160
4	Afghanistan	2013	Céréales	40504

```
aide_alimentaire = aide_alimentaire.rename(columns={'Pays bénéficiaire': 'Zone'})

aide_alimentaire['Valeur']*=1000

aide_alimentaire.head(5)
```

	Zone	Année	Produit	Valeur
0	Afghanistan	2013	Autres non-céréales	682000
1	Afghanistan	2014	Autres non-céréales	335000
2	Afghanistan	2013	Blé et Farin	39224000
3	Afghanistan	2014	Blé et Farin	15160000
4	Afghanistan	2013	Céréales	40504000

2.4 - Analyse exploratoire du fichier sous nutrition

```
print("Le tableau comporte {} observation(s) ou article(s)".format(sous_nutrition.shape[0]))
print("Le tableau comporte {} colonne(s)".format(sous_nutrition.shape[1]))
```

```
Le tableau comporte 1218 observation(s) ou article(s)
Le tableau comporte 3 colonne(s)
```

```
sous_nutrition.shape[1]
```

```
3
```

```
sous_nutrition.head(5)
```

	Zone	Année	Valeur
0	Afghanistan	2012-2014	8.6
1	Afghanistan	2013-2015	8.8
2	Afghanistan	2014-2016	8.9
3	Afghanistan	2015-2017	9.7
4	Afghanistan	2016-2018	10.5

```
sous_nutrition.dtypes
```

```
Zone      object
Année     object
Valeur    object
dtype: object
```

```
sous_nutrition['Valeur'] = pd.to_numeric(sous_nutrition['Valeur'], errors='coerce')

sous_nutrition = sous_nutrition.fillna(0)

sous_nutrition= sous_nutrition.rename(columns={'Valeur' : 'Sous_nutrition'})

sous_nutrition['Sous_nutrition'] *= 1000000

sous_nutrition.head(5)
```

	Zone	Année	Sous_nutrition
0	Afghanistan	2012-2014	8600000.0
1	Afghanistan	2013-2015	8800000.0
2	Afghanistan	2014-2016	8900000.0
3	Afghanistan	2015-2017	9700000.0
4	Afghanistan	2016-2018	10500000.0

3.1 - Proportion de personnes en sous nutrition

```
# Filtre pour sélectionner uniquement les données de l'année 2017
population_2017 = population[population['Année'] == 2017]
sous_nutrition_2017 = sous_nutrition[sous_nutrition['Année'] == '2016-2018']

# Jointure des deux tables en utilisant la colonne "Zone" comme clé de jointure
jointure = pd.merge(population_2017, sous_nutrition_2017, on='Zone')

# Affichage des résultats de la jointure
print(jointure)
```

```
                                        Zone  Année_x  Population    Année_y  \
0                                Afghanistan     2017  36296113.0  2016-2018
1                           Afrique du Sud       2017  57009756.0  2016-2018
2                                  Albanie      2017   2884169.0  2016-2018
3                                  Algérie      2017  41389189.0  2016-2018
4                                 Allemagne     2017  82658409.0  2016-2018
..                                      ...      ...         ...         ...
198  Venezuela (République bolivarienne du)     2017  29402484.0  2016-2018
199                              Viet Nam      2017  94600648.0  2016-2018
200                                 Yémen      2017  27834819.0  2016-2018
201                                Zambie      2017  16853599.0  2016-2018
202                               Zimbabwe     2017  14236595.0  2016-2018

     Sous_nutrition
0       10500000.0
1        3100000.0
2         100000.0
```

```
3         1300000.0
4               0.0
..              ...
198       8000000.0
199       6500000.0
200             0.0
201             0.0
202             0.0

[203 rows x 5 columns]
```

```
nombre_sous_nutrition = jointure['Sous_nutrition'].sum()
```

```
print("Nombre de personnes en état de sous-nutrition en 2017 :", nombre_sous_nutrition)
```

```
Nombre de personnes en état de sous-nutrition en 2017 : 535700000.0
```

```
# Calcul de la population totale
population_totale = population_2017['Population'].sum()

# Calcul de la proportion de personne en sous-nutrition
proportion_sous_nutrition = (nombre_sous_nutrition / population_totale) * 100
# Affichage du résultat
print("Proportion de personnes en sous-nutrition en 2017 : {:.2f}%".format(proportion_sous_nutrition))
```

```
Proportion de personnes en sous-nutrition en 2017 : 7.10%
```

3.2 - Nombre théorique de personne qui pourrait être nourries

```
# Jointure des deux dataframes en utilisant la colonne "Zone" comme clé de jointure
dispo_alimentaire_population = pd.merge(dispo_alimentaire, population, on='Zone')

# Affichage du dataframe résultant avec la population ajoutée
print(dispo_alimentaire_population)
```

```
                Zone           Produit   Origine  Aliments pour animaux  \
0        Afghanistan  Abats Comestible   animale                    0.0
1        Afghanistan  Abats Comestible   animale                    0.0
2        Afghanistan  Abats Comestible   animale                    0.0
3        Afghanistan  Abats Comestible   animale                    0.0
4        Afghanistan  Abats Comestible   animale                    0.0
...              ...               ...       ...                    ...
92491  Îles Salomon    Épices, Autres  vegetale                    0.0
92492  Îles Salomon    Épices, Autres  vegetale                    0.0
92493  Îles Salomon    Épices, Autres  vegetale                    0.0
92494  Îles Salomon    Épices, Autres  vegetale                    0.0
92495  Îles Salomon    Épices, Autres  vegetale                    0.0

       Autres Utilisations  Disponibilité alimentaire (Kcal/personne/jour)  \
0                      0.0                                             5.0
1                      0.0                                             5.0
2                      0.0                                             5.0
3                      0.0                                             5.0
4                      0.0                                             5.0
...                    ...                                             ...
92491                  0.0                                             4.0
92492                  0.0                                             4.0
92493                  0.0                                             4.0
92494                  0.0                                             4.0
92495                  0.0                                             4.0

       Disponibilité alimentaire en quantité (kg/personne/an)  \
0                                                   1.72
1                                                   1.72
2                                                   1.72
3                                                   1.72
4                                                   1.72
...                                                  ...
92491                                               0.48
92492                                               0.48
92493                                               0.48
92494                                               0.48
92495                                               0.48

       Disponibilité de matière grasse en quantité (g/personne/jour)  \
0                                                   0.20
1                                                   0.20
2                                                   0.20
3                                                   0.20
4                                                   0.20
...                                                  ...
92491                                               0.21
92492                                               0.21
92493                                               0.21
92494                                               0.21
92495                                               0.21

       Disponibilité de protéines en quantité (g/personne/jour)  \
0                                                   0.77
1                                                   0.77
2                                                   0.77
3                                                   0.77
4                                                   0.77
```

```
# Calcul de la disponibilité alimentaire en kilocalories pour chaque produit et chaque zone
dispo_alimentaire_population['dispo_kcal'] = dispo_alimentaire_population['Disponibilité alimentaire (Kcal/personne/jour)'] * dispo_alimentaire_population['Disponibilité alimentaire en quantité (kg/personne/an)'] * dispo_alimentaire_population['Population'] * 365

# Agrégation des données pour obtenir la disponibilité alimentaire mondiale en kilocalories par zone
dispo_kcal_mondiale = dispo_alimentaire_population.groupby('Zone')['dispo_kcal'].sum().reset_index()

# Jointure pour ajouter la colonne dispo_kcal au dataframe dispo_alimentaire_population
dispo_alimentaire_population = pd.merge(dispo_alimentaire_population, dispo_kcal_mondiale, on='Zone', suffixes=('', '_mondiale'))

# Affichage du dataframe avec la colonne dispo_kcal
print(dispo_alimentaire_population)
```

```
                Zone           Produit   Origine  Aliments pour animaux  \
0        Afghanistan  Abats Comestible   animale                    0.0
1        Afghanistan  Abats Comestible   animale                    0.0
2        Afghanistan  Abats Comestible   animale                    0.0
3        Afghanistan  Abats Comestible   animale                    0.0
4        Afghanistan  Abats Comestible   animale                    0.0
...              ...               ...       ...                    ...
92491  Îles Salomon    Épices, Autres  vegetale                    0.0
92492  Îles Salomon    Épices, Autres  vegetale                    0.0
92493  Îles Salomon    Épices, Autres  vegetale                    0.0
92494  Îles Salomon    Épices, Autres  vegetale                    0.0
92495  Îles Salomon    Épices, Autres  vegetale                    0.0

       Autres Utilisations  Disponibilité alimentaire (Kcal/personne/jour)  \
0                      0.0                                             5.0
1                      0.0                                             5.0
2                      0.0                                             5.0
3                      0.0                                             5.0
4                      0.0                                             5.0
...                    ...                                             ...
92491                  0.0                                             4.0
92492                  0.0                                             4.0
92493                  0.0                                             4.0
92494                  0.0                                             4.0
92495                  0.0                                             4.0

       Disponibilité alimentaire en quantité (kg/personne/an)  \
0                                                   1.72
1                                                   1.72
2                                                   1.72
3                                                   1.72
4                                                   1.72
...                                                  ...
92491                                               0.48
92492                                               0.48
92493                                               0.48
92494                                               0.48
92495                                               0.48

       Disponibilité de matière grasse en quantité (g/personne/jour)  \
0                                                   0.20
1                                                   0.20
2                                                   0.20
3                                                   0.20
4                                                   0.20
...                                                  ...
92491                                               0.21
92492                                               0.21
92493                                               0.21
92494                                               0.21
92495                                               0.21

       Disponibilité de protéines en quantité (g/personne/jour)  \
0                                                   0.77
1                                                   0.77
2                                                   0.77
3                                                   0.77
4                                                   0.77
```

```
# Filtrez les données pour l'année 2017
dispo_alimentaire_2017 = dispo_alimentaire_population[dispo_alimentaire_population['Année'] == 2017]


dispo_en_kcl_mondial_2017= dispo_alimentaire_2017['dispo_kcal_mondiale'].sum()
print(dispo_en_kcl_mondial_2017)
```

```
2.4813710771098118e+20
```

```
# Consommation moyenne de kilocalories par jour par personne
consommation_moyenne_kcal = 2250 # Exemple : consommation moyenne recommandée


# Consommation moyenne de kilocalories par jour par la population totale en 2017
Consommation_moyenne_kcal_mondiale_2017 = consommation_moyenne_kcal * population_totale
print("Consommation moyenne kcal mondiale 2017:", Consommation_moyenne_kcal_mondiale_2017)
```

```
Consommation moyenne kcal mondiale 2017: 16983301749750.0
```

```
# Calcul du nombre d'humains pouvant être nourris
nombre_humains_nourris = (dispo_en_kcl_mondial_2017 / Consommation_moyenne_kcal_mondiale_2017) * 365


# Affichage du résultat
print("Nombre d'humains pouvant être nourris :", round(nombre_humains_nourris))
```

```
Nombre d'humains pouvant être nourris : 5332887895
```

3.3 - Nombre théorique de personne qui pourrait être nourrie avec les produits végétaux

```
# Filtrez les données pour l'année 2017
dispo_alimentaire_2017 = dispo_alimentaire_population[dispo_alimentaire_population['Année'] == 2017]

# Filtrez les données pour ne garder que les produits d'origine végétale
dispo_vegetaux_2017 = dispo_alimentaire_2017[dispo_alimentaire_2017['Origine'] == 'vegetale']


# Calcul de la disponibilité alimentaire totale en kilocalories pour les végétaux
kcal_disponibles_vegetaux = dispo_vegetaux_2017['dispo_kcal_mondiale'].sum()

# Affichage du résultat
print("Nombre de kilocalories disponibles pour les végétaux :", kcal_disponibles_vegetaux)
```

```
Nombre de kilocalories disponibles pour les végétaux : 1.9059029786981686e+20
```

```
# Consommation moyenne de kilocalories par jour par personne
consommation_moyenne_kcal = 2250 # Exemple : consommation moyenne recommandée

# Calcul du nombre d'humains pouvant être nourris avec les végétaux
nombre_humains_nourris_vegetaux = (kcal_disponibles_vegetaux / Consommation_moyenne_kcal_mondiale_2017) * 365

# Affichage du résultat
print("Nombre d'humains pouvant être nourris avec les végétaux :", int(nombre_humains_nourris_vegetaux))
```

```
Nombre d'humains pouvant être nourris avec les végétaux : 4096109210
```

3.4 - Utilisation de la disponibilité intérieure

```
# Filtrez les données pour l'année 2017
data_2017 = dispo_alimentaire_population[dispo_alimentaire_population['Année'] == 2017]

# Calculez la somme de la disponibilité intérieure pour l'année 2017
disponibilite_intérieur_2017 = data_2017['Disponibilité intérieure'].sum()
print("La disponibilité intérieure pour l'année 2017 est :", disponibilite_intérieur_2017)

# Colonnes à afficher
colonnes_a_afficher = ['Aliments pour animaux', 'Pertes', 'Nourriture', 'Semences', 'Traitement', 'Autres Utilisations']

# Boucle pour afficher les totaux des colonnes spécifiées
for colonne in colonnes_a_afficher:
    total_colonne = data_2017[colonne].sum()
    print("Total de la colonne", colonne, "pour l'année 2017 :", total_colonne)
```

```
 La disponibilité intérieure pour l'année 2017 est : 9733927000.0
 Total de la colonne Aliments pour animaux pour l'année 2017 : 1288002.0
 Total de la colonne Pertes pour l'année 2017 : 452283000.0
 Total de la colonne Nourriture pour l'année 2017 : 4805525000.0
 Total de la colonne Semences pour l'année 2017 : 153317000.0
 Total de la colonne Traitement pour l'année 2017 : 2185641000.0
 Total de la colonne Autres Utilisations pour l'année 2017 : 858771000.0
```

```
# Calcul de la somme des aliments pour animaux
somme_aliments_animaux = data_2017['Aliments pour animaux'].sum()

# Calcul de la somme de la disponibilité alimentaire totale
somme_dispo_alimentaire = data_2017['Disponibilité intérieure'].sum()

# Calcul de la proportion d'alimentation animale
proportion_alimentation_animale = (somme_aliments_animaux / somme_dispo_alimentaire) * 100

# Affichage de la proportion d'alimentation animale
print("Proportion d'alimentation animale : {:.2f}%".format(proportion_alimentation_animale))
```

```
Proportion d'alimentation animale : 0.01%
```

3.5 - Utilisation des céréales

```
# Liste des céréales spécifiques
cereales = ['Blé', 'Riz (Eq blanchi)', 'Maïs','Orge', 'Seigle', 'Avoine', 'Millet', 'Sorgho', 'Céréales', 'Autres']
# Filtrage du dataframe pour ne garder que les lignes correspondant aux céréales spécifiques
cereales_dataframe = dispo_alimentaire[dispo_alimentaire['Produit'].isin(cereales)]


# Calcul de la somme de la disponibilité alimentaire pour chaque céréale spécifique
somme_dispo_total = cereales_dataframe.groupby('Produit')['Disponibilité alimentaire en quantité (kg/personne/an)'].sum()

# Filtrage pour ne garder que les lignes où les aliments sont utilisés pour l'alimentation animale
alimentation_animale = cereales_dataframe[cereales_dataframe['Aliments pour animaux'] > 0 ]

# Calcul de la somme de la disponibilité alimentaire pour chaque céréale spécifique utilisée pour l'alimentation animale
somme_dispo_animale = alimentation_animale.groupby('Produit')['Disponibilité alimentaire en quantité (kg/personne/an)'].sum()

# Calcul de la part d'alimentation animale en pourcentage pour chaque céréale spécifique
part_alimentation_animale_pourcentage = round((somme_dispo_animale / somme_dispo_total) * 100, 2)

# Affichage des résultats
print(part_alimentation_animale_pourcentage)
```

```
Produit
Avoine    65.61
Blé       70.48
Maïs      91.06
```

```
Millet    75.24
Orge      75.51
Seigle    97.30
Sorgho    72.16
Name: Disponibilité alimentaire en quantité (kg/personne/an), dtype: float64
```

```
# Calcul de la somme de la disponibilité alimentaire de toutes les céréales spécifiques
somme_dispo_total = cereales_dataframe['Disponibilité alimentaire en quantité (kg/personne/an)'].sum()

# Filtrage pour ne garder que les lignes où les aliments sont utilisés pour l'alimentation animale
alimentation_animale = cereales_dataframe[cereales_dataframe['Aliments pour animaux'] > 0]

# Calcul de la somme de la disponibilité alimentaire destinée à l'alimentation animale de toutes les céréales spécifiques
somme_dispo_animale = alimentation_animale['Disponibilité alimentaire en quantité (kg/personne/an)'].sum()

# Calcul de la part totale destinée à l'alimentation animale en pourcentage
part_alimentation_animale_pourcentage = (somme_dispo_animale / somme_dispo_total) * 100

# Affichage du résultat
print("La part totale destinée à l'alimentation animale en pourcentage : {:.2f}%".format(part_alimentation_animale_pourcentage))
```

```
La part totale destinée à l'alimentation animale en pourcentage : 75.36%
```

```
# Filtrage pour ne garder que les lignes où les aliments sont destinés à l'alimentation humaine (nourriture)
nourriture_humaine = cereales_dataframe[cereales_dataframe['Aliments pour animaux'] == 0]

# Calcul de la somme totale de la disponibilité alimentaire destinée à l'alimentation humaine
somme_dispo_nourriture_total = nourriture_humaine['Disponibilité alimentaire en quantité (kg/personne/an)'].sum()

# Calcul de la part d'utilisation humaine (nourriture) en pourcentage pour chaque céréale spécifique
part_nourriture_pourcentage = round((nourriture_humaine.groupby('Produit')['Disponibilité alimentaire en quantité (kg/personne/an)'].sum() / somme_dispo_nourriture_total) * 100, 2)

# Affichage des résultats
print(part_nourriture_pourcentage)
```

```
Produit
Avoine     1.19
Blé       82.89
Maïs       7.51
Millet     2.89
Orge       1.47
Seigle     0.16
Sorgho     3.89
Name: Disponibilité alimentaire en quantité (kg/personne/an), dtype: float64
```

```
# Filtrage pour ne garder que les lignes où les aliments sont destinés à la nourriture humaine
nourriture_humaine = cereales_dataframe[cereales_dataframe['Aliments pour animaux'] == 0]

# Calcul de la somme de la disponibilité alimentaire destinée à la nourriture pour toutes les céréales spécifiques
somme_dispo_nourriture = nourriture_humaine['Disponibilité alimentaire en quantité (kg/personne/an)'].sum()

# Calcul de la part totale destinée à la nourriture en pourcentage
part_nourriture_pourcentage = (somme_dispo_nourriture / somme_dispo_total) * 100

# Affichage du résultat
print("La part totale destinée à la nourriture en pourcentage : {:.2f}%".format(part_nourriture_pourcentage))
```

```
La part totale destinée à la nourriture en pourcentage : 24.64%
```

3.6 - Pays avec la proportion de personnes sous-alimentée la plus forte en 2017

```
# Calcul de la somme de sous-nutrition par pays
somme_sous_nutrition = sous_nutrition.groupby('Zone')['Sous_nutrition'].sum()

# Création de la colonne "proportion"
sous_nutrition['proportion'] = (sous_nutrition['Sous_nutrition'] / sous_nutrition['Zone'].map(somme_sous_nutrition)).round(2)


# Filtrage des données pour l'intervalle 2016-2018
sous_nutrition_2017 = sous_nutrition[sous_nutrition['Année'].str.contains('2016-2018')]

# Calcul de la moyenne de sous-nutrition sur les 3 années
df_mean = sous_nutrition_2017.groupby('Zone')['Sous_nutrition']
# Triage des données en fonction de la proportion de manière décroissante
proportion_sous_nutrition = sous_nutrition_2017.sort_values(by='proportion', ascending=False)
# Sélection des 10 premiers pays avec la proportion la plus élevée
top_10_pays =proportion_sous_nutrition.head(10)

# Affichage des 10 premiers pays avec la proportion la plus élevée
print("Les 10 pays avec la proportion la plus élevée de personnes sous-alimentées en 2017 :")
print(top_10_pays[['Zone', 'Sous_nutrition', 'proportion']])
```

```
Les 10 pays avec la proportion la plus élevée de personnes sous-alimentées en 2017 :
                                    Zone  Sous_nutrition  proportion
1174                             Ukraine       1500000.0        0.34
1192  Venezuela (République bolivarienne du)       8000000.0        0.25
58                             Argentine       1500000.0        0.22
604                              Lesotho        800000.0        0.21
412                              Géorgie        300000.0        0.21
406                               Gambie        300000.0        0.20
706                              Mexique       8400000.0        0.20
700                           Mauritanie        500000.0        0.20
772                              Nigéria      22800000.0        0.19
730                           Mozambique       9400000.0        0.19
```

3.7 - Pays qui ont le plus bénéficié d'aide alimentaire depuis 2013

```
aide_alimentaire.head()
```

	Zone	Année	Produit	Valeur
0	Afghanistan	2013	Autres non-céréales	682000
1	Afghanistan	2014	Autres non-céréales	335000
2	Afghanistan	2013	Blé et Farin	39224000
3	Afghanistan	2014	Blé et Farin	15160000
4	Afghanistan	2013	Céréales	40504000

```
# Calcul du total de l'aide alimentaire par pays
total_aide_par_pays = aide_alimentaire.groupby('Zone')['Valeur'].sum()

# Affichage du total de l'aide alimentaire par pays
print(total_aide_par_pays)
```

```
Zone
Afghanistan     185452000
Algérie          81114000
Angola            5014000
Bangladesh      348188000
Bhoutan           2666000
                   ...
Zambie            3026000
Zimbabwe         62570000
Égypte            1122000
Équateur          1362000
Éthiopie       1381294000
Name: Valeur, Length: 76, dtype: int64
```

```
# Filtrage des données pour ne conserver que les lignes à partir de l'année 2013
aide_alimentaire_depuis_2013 = aide_alimentaire[aide_alimentaire['Année'] >= 2013]

# Calcul de la somme de l'aide alimentaire par pays
total_aide_par_pays = aide_alimentaire_depuis_2013.groupby('Zone')['Valeur'].sum()

# Triage des données en fonction de la somme de manière décroissante
total_aide_par_pays = total_aide_par_pays.sort_values(ascending=False)

# Sélection des 10 premiers pays
top_10_pays =total_aide_par_pays.head(10)

# Affichage des 10 premiers pays qui ont bénéficié le plus de l'aide alimentaire depuis 2013
print("Les 10 pays qui ont bénéficié le plus de l'aide alimentaire depuis 2013 :")
print(top_10_pays)
```

```
Les 10 pays qui ont bénéficié le plus de l'aide alimentaire depuis 2013 :
Zone
République arabe syrienne            1858943000
Éthiopie                             1381294000
Yémen                                1206484000
Soudan du Sud                         695248000
Soudan                                669784000
Kenya                                 552836000
Bangladesh                            348188000
Somalie                               292678000
République démocratique du Congo      288502000
Niger                                 276344000
Name: Valeur, dtype: int64
```

3.8 - Evolution des 5 pays qui ont le plus bénéficiés de l'aide alimentaire entre 2013 et 2016

```
# Calcul du total de l'aide alimentaire par pays
total_aide_par_pays = aide_alimentaire.groupby('Zone')['Valeur'].sum()

# Triage des données en fonction de la somme de manière décroissante
total_aide_trie = total_aide_par_pays.sort_values(ascending=False)

# Sélection des 5 premiers pays
top_5_pays = total_aide_trie.head(5)

# Création de la liste des 5 pays les plus bénéficiaires d'aide alimentaire
liste_pays = top_5_pays.index.tolist()

# Affichage de la liste des 5 pays
print("Les 5 pays les plus bénéficiaires d'aide alimentaire :")
print(liste_pays)
```

```
Les 5 pays les plus bénéficiaires d'aide alimentaire :
['République arabe syrienne', 'Éthiopie', 'Yémen', 'Soudan du Sud', 'Soudan']
```

```
# Filtrage du dataframe initial en utilisant la liste des pays bénéficiaires
df_filtre = aide_alimentaire[aide_alimentaire['Zone'].isin(['République arabe syrienne', 'Éthiopie', 'Yémen', 'Soudan du Sud', 'Soudan'])]



# Filtrer les données pour la période de 2013 à 2016
donnees_periode = df_filtre[(df_filtre['Année'] >= 2013) & (df_filtre['Année'] <= 2016)]

# Regrouper les données par pays et année, puis calculer la somme de l'aide alimentaire
evolution_aide_alimentaire = donnees_periode.groupby(['Zone', 'Année'])['Valeur'].sum()

# Afficher l'évolution de l'aide alimentaire pour chaque pays
for pays in liste_pays:
    evolution_pays = evolution_aide_alimentaire[pays]
    print(f"Évolution de l'aide alimentaire pour {pays}:")
    print(evolution_pays)
    print()
```

```
Évolution de l'aide alimentaire pour République arabe syrienne:
Année
2013    563566000
2014    651870000
2015    524949000
2016    118558000
Name: Valeur, dtype: int64

Évolution de l'aide alimentaire pour Éthiopie:
Année
2013    591404000
2014    586624000
2015    203266000
Name: Valeur, dtype: int64

Évolution de l'aide alimentaire pour Yémen:
Année
2013    264764000
2014    103840000
2015    372306000
2016    465574000
Name: Valeur, dtype: int64

Évolution de l'aide alimentaire pour Soudan du Sud:
Année
2013    196330000
2014    450610000
2015     48308000
Name: Valeur, dtype: int64

Évolution de l'aide alimentaire pour Soudan:
Année
2013    330230000
2014    321904000
2015     17650000
Name: Valeur, dtype: int64
```

3.9 - Pays avec le moins de disponibilité par habitant

```
# Groupement par pays et calcul de la disponibilité alimentaire moyenne par personne
dispo_alimentaire_moyenne = dispo_alimentaire.groupby('Zone')['Disponibilité alimentaire (Kcal/personne/jour)'].mean()

# Tri des pays par ordre croissant de disponibilité alimentaire moyenne
pays_moins_dispo = round(dispo_alimentaire_moyenne.nsmallest(10), 2)

# Affichage des 10 pays qui ont le moins de disponibilité alimentaire par personne
print("Les 10 pays qui ont le moins de disponibilité alimentaire par personne :")
print(pays_moins_dispo)
```

```
Les 10 pays qui ont le moins de disponibilité alimentaire par personne :
Zone
Zambie                          20.69
République centrafricaine       22.11
Madagascar                      22.35
Ouganda                         22.38
République-Unie de Tanzanie     22.96
Zimbabwe                        22.97
Kenya                           22.97
Namibie                         23.04
Éthiopie                        23.14
Yémen                           23.84
Name: Disponibilité alimentaire (Kcal/personne/jour), dtype: float64
```

3.10 - Pays avec le plus de disponibilité par habitant

```
# Groupement par pays et calcul de la disponibilité alimentaire moyenne par personne
dispo_alimentaire_moyenne = dispo_alimentaire.groupby('Zone')['Disponibilité alimentaire (Kcal/personne/jour)'].mean()

# Tri des pays par ordre décroissant de disponibilité alimentaire moyenne
pays_plus_dispo = round(dispo_alimentaire_moyenne.nlargest(10), 2)

# Affichage des 10 pays qui ont le plus de disponibilité alimentaire par personne en 2017
print("Les 10 pays qui ont le plus de disponibilité alimentaire par personne en 2017 :")
print(pays_plus_dispo)
```

```
Les 10 pays qui ont le plus de disponibilité alimentaire par personne en 2017 :
Zone
Turkménistan    49.75
```

```
Lesotho         47.70
Kiribati        41.67
Turquie         39.87
Autriche        39.68
Israël          39.67
Belgique        39.34
Monténégro      39.21
Samoa           38.89
Koweït          38.88
Name: Disponibilité alimentaire (Kcal/personne/jour), dtype: float64
```

3.11 - Exemple de la Thaïlande pour le Manioc

```
# Création du nouveau dataframe avec uniquement les données de la Thaïlande
df_thailande = sous_nutrition.loc[sous_nutrition['Zone'] == 'Thaïlande']


# Filtrer les données pour obtenir celles de la Thaïlande en 2017
sous_nutrition_thailande_2017 = df_thailande[df_thailande['Année'] == '2016-2018']['Sous_nutrition'].values[0]

print("Nombre de personnes en sous-nutrition en Thaïlande en 2017:", sous_nutrition_thailande_2017)
```

```
Nombre de personnes en sous-nutrition en Thaïlande en 2017: 6200000.0
```

```
# Filtrer les lignes pour ne garder que la disponibilité alimentaire et le produit Manioc en Thailande
thailande_2017 = dispo_alimentaire[(dispo_alimentaire['Zone'] == 'Thaïlande') & (dispo_alimentaire['Produit'] == 'Manioc')]

# Extraire la valeur "Disponibilité alimentaire (Kcal/personne/jour)"
kcal_par_personne_par_jour = thailande_2017['Disponibilité alimentaire (Kcal/personne/jour)']

print(f"Disponibilité alimentaire en kcal de Manioc en Thaïlande par personne en 2017 : {kcal_par_personne_par_jour}")
```

```
Disponibilité alimentaire en kcal de Manioc en Thaïlande par personne en 2017 : 13809    40.0
Name: Disponibilité alimentaire (Kcal/personne/jour), dtype: float64
```

```
# Filtrer les données pour "produit" = "Manioc"
manioc_2017 = dispo_alimentaire[dispo_alimentaire["Produit"] == "Manioc"]

# Calculer la production totale de manioc en 2017
production_totale_2017 = manioc_2017["Production"].sum()

print("Production totale de manioc en 2017 :", production_totale_2017)
```

```
Production totale de manioc en 2017 : 255002000.0
```

```
# Calculer la production totale de manioc en Thailand
production_totale_manioc_2017 = thailande_2017['Production'].sum()

print("Production totale de Manioc en 2017 :", production_totale_manioc_2017)
```

```
Production totale de Manioc en 2017 : 30228000.0
```

```
# Production totale mondiale de manioc en 2017
production_mondiale = 255002000.0

# Calculer la proportion de la production de Thaïlande par rapport à la production mondiale
proportion = round((production_totale_manioc_2017 / production_mondiale) * 100, 2)

print("Proportion de la production de Thaïlande par rapport à la production mondiale en 2017:", proportion, "%")
```

```
Proportion de la production de Thaïlande par rapport à la production mondiale en 2017: 11.85 %
```

```
# Calculer la somme des exportations de manioc en Thaïlande en 2017
exportations_manioc_thailande = thailande_2017['Exportations - Quantité'].sum()

# Calculer la production totale de manioc en Thaïlande en 2017
production_totale_manioc_thailande = thailande_2017['Production'].sum()

# Calculer le pourcentage d'exportation de manioc en Thaïlande en 2017
pourcentage_export_manioc_thailande = round((exportations_manioc_thailande / production_totale_manioc_thailande) * 100, 2)

print("Pourcentage d'exportation de Manioc en Thaïlande en 2017 :", pourcentage_export_manioc_thailande)
```

```
Pourcentage d'exportation de Manioc en Thaïlande en 2017 : 83.41
```

Analyses complémentaires

la proportion de la sous nutrition en thailand de 2013 à 2018

```
# Filtrer les lignes pour la Thaïlande
thailande_data = sous_nutrition[sous_nutrition['Zone'] == 'Thaïlande']

# Triez les données par année pour s'assurer qu'elles sont dans l'ordre chronologique
thailande_data = thailande_data.sort_values(by='Année')

# Affichez l'évolution de la sous-nutrition en Thaïlande au fil des années
print(thailande_data)
```

```
            Zone      Année  Sous_nutrition  proportion
1110  Thaïlande  2012-2014       6200000.0        0.17
1111  Thaïlande  2013-2015       6000000.0        0.16
1112  Thaïlande  2014-2016       5900000.0        0.16
1113  Thaïlande  2015-2017       6000000.0        0.16
1114  Thaïlande  2016-2018       6200000.0        0.17
1115  Thaïlande  2017-2019       6500000.0        0.18
```

les origines de produits

```
# Grouper les données filtrées par origine, puis compter le nombre de produits de chaque origine pour l'année 2017
origin_counts_2017 = dispo_alimentaire_2017.groupby('Origine')['Produit'].count()

# Afficher les chiffres pour l'année 2017
print(origin_counts_2017)
```

```
Origine
animale      3665
vegetale    11751
Name: Produit, dtype: int64
```

Dispo_alimentaire en kcal des 10 pays les plus sous-alimentés en 2017

```
# Pays à calculer
pays_a_calculer = ['Ukraine', 'Venezuela (République bolivarienne du)', 'Argentine', 'Lesotho', 'Géorgie', 'Gambie', 'Mexique', 'Mauritanie', 'Nigéria','Mozambique']
# Calcul de la moyenne de la disponibilité en kcal pour chaque pays spécifié
moyennes_par_pays = dispo_alimentaire[dispo_alimentaire['Zone'].isin(pays_a_calculer)].groupby('Zone')['Disponibilité alimentaire (Kcal/personne/jour)'].mean()

# Affichage des moyennes pour chaque pays
for pays, moyenne in moyennes_par_pays.items():
    print(f"Moyenne de la disponibilité en kcal pour {pays}: {moyenne:.2f} kcal")
```

```
Moyenne de la disponibilité en kcal pour Argentine: 34.69 kcal
Moyenne de la disponibilité en kcal pour Gambie: 30.13 kcal
Moyenne de la disponibilité en kcal pour Géorgie: 31.89 kcal
Moyenne de la disponibilité en kcal pour Lesotho: 47.70 kcal
Moyenne de la disponibilité en kcal pour Mauritanie: 31.94 kcal
Moyenne de la disponibilité en kcal pour Mexique: 32.29 kcal
Moyenne de la disponibilité en kcal pour Mozambique: 25.64 kcal
Moyenne de la disponibilité en kcal pour Nigéria: 29.67 kcal
Moyenne de la disponibilité en kcal pour Ukraine: 34.48 kcal
Moyenne de la disponibilité en kcal pour Venezuela (République bolivarienne du): 27.43 kcal
```

Disponibilité alimentaire des 10 pays les plus peuplés en 2017

```
# Sélectionner les données pour l'année 2017
population_2017 = population[population['Année'] == 2017]

# Trier les données par population de manière décroissante et sélectionner les 10 zones les plus peuplées
top_10_zones = population_2017.sort_values(by='Population', ascending=False).head(10)

# Affichage des 10 zones les plus peuplées en 2017
print(top_10_zones)
```

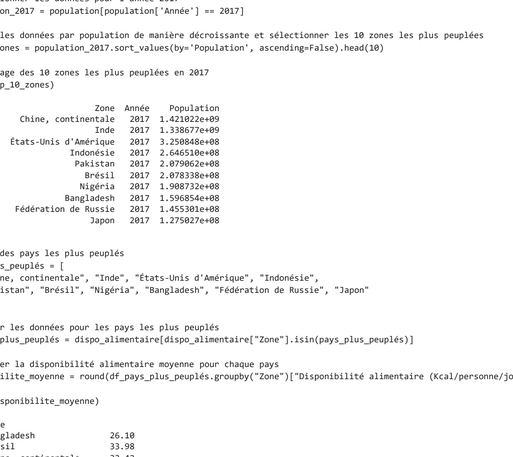
```
                        Zone  Année    Population
262    Chine, continentale   2017  1.421022e+09
622                   Inde   2017  1.338677e+09
394  États-Unis d'Amérique   2017  3.250848e+08
628             Indonésie   2017  2.646510e+08
964              Pakistan   2017  2.079062e+08
190                Brésil   2017  2.078338e+08
916               Nigéria   2017  1.908732e+08
118            Bangladesh   2017  1.596854e+08
406   Fédération de Russie   2017  1.455301e+08
676                 Japon   2017  1.275027e+08
```

```
# Liste des pays les plus peuplés
pays_plus_peuplés = [
    "Chine, continentale", "Inde", "États-Unis d'Amérique", "Indonésie",
    "Pakistan", "Brésil", "Nigéria", "Bangladesh", "Fédération de Russie", "Japon"
]

# Filtrer les données pour les pays les plus peuplés
df_pays_plus_peuplés = dispo_alimentaire[dispo_alimentaire["Zone"].isin(pays_plus_peuplés)]

# Calculer la disponibilité alimentaire moyenne pour chaque pays
disponibilite_moyenne = round(df_pays_plus_peuplés.groupby("Zone")["Disponibilité alimentaire (Kcal/personne/jour)"].mean(), 2)

print(disponibilite_moyenne)
```

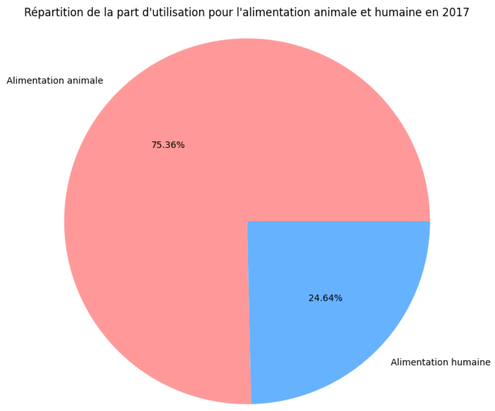
```
Zone
Bangladesh                26.10
Brésil                    33.98
Chine, continentale       32.42
Fédération de Russie      35.74
Inde                      25.30
Indonésie                 28.92
Japon                     28.10
Nigéria                   29.67
Pakistan                  25.94
États-Unis d'Amérique     38.76
Name: Disponibilité alimentaire (Kcal/personne/jour), dtype: float64
```

GRAPHIQUES

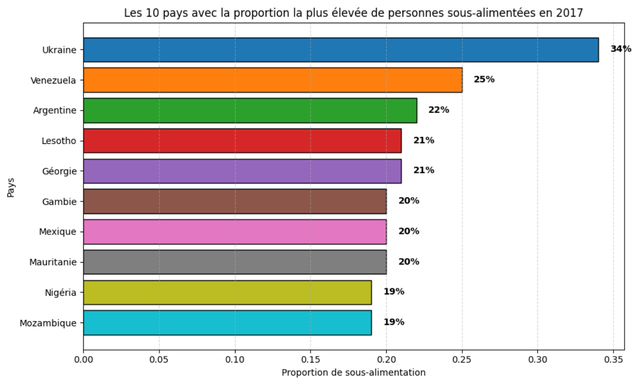
```
import matplotlib.pyplot as plt


# Noms des colonnes
colonnes = ['Aliments pour animaux', 'Pertes', 'Nourriture', 'Semences', 'Traitement', 'Autres Utilisations']

# Totaux des colonnes correspondant à l'année 2017
totaux = [1288002.0, 452283000.0, 4805525000.0, 153317000.0, 2185641000.0, 858771000.0]

# Modifier la part d'utilisation pour "Aliments pour animaux" à 0.01%
totaux[0] = 0.01

# Création du graphique en camembert
plt.figure(figsize=(8, 8))
plt.pie(totaux, labels=colonnes, autopct='%1.1f%%', colors=['skyblue', 'lightgreen', 'lightcoral', 'lightyellow', 'lightpink', 'lightblue'])

# Ajout de titre
plt.title('Répartition des totaux des différentes colonnes pour l\'année 2017')

# Affichage du graphique
plt.axis('equal')
plt.show()
```



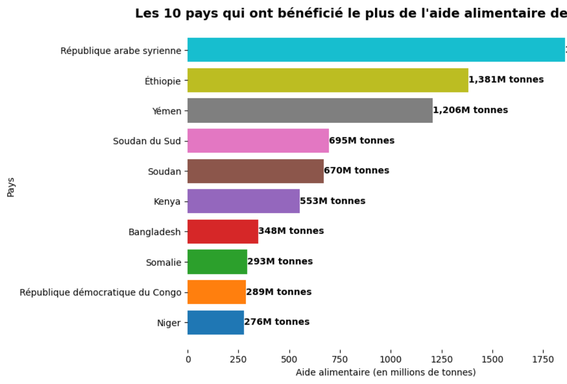

```
# Pourcentages
part_alimentation_animale = 75.36
part_alimentation_humaine = 100 - part_alimentation_animale

# Noms des sections
sections = ['Alimentation animale', 'Alimentation humaine']

# Données pour chaque section
data = [part_alimentation_animale, part_alimentation_humaine]

# Couleurs des sections
colors = ['#FF9999', '#66B2FF']  # Rouge pour animale, Bleu pour humaine

# Création du graphique en camembert avec couleurs différentes
plt.figure(figsize=(8, 8))
plt.pie(data, labels=sections, autopct='%1.2f%%', colors=colors)

# Ajout de titre
plt.title('Répartition de la part d\'utilisation pour l\'alimentation animale et humaine en 2017')

# Affichage du graphique
plt.axis('equal')
plt.show()
```



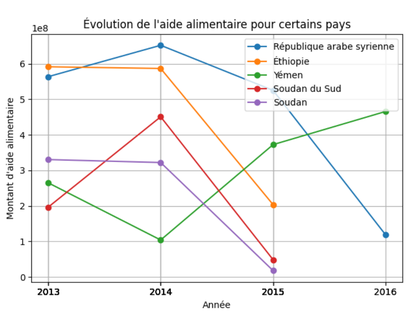

```
# Données des pays et des proportions de sous-alimentation
pays = ['Ukraine', 'Venezuela', 'Argentine', 'Lesotho', 'Géorgie', 'Gambie', 'Mexique', 'Mauritanie', 'Nigéria', 'Mozambique']
proportions = [0.34, 0.25, 0.22, 0.21, 0.21, 0.20, 0.20, 0.20, 0.19, 0.19]

# Couleurs personnalisées pour les barres
couleurs = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd', '#8c564b', '#e377c2', '#7f7f7f', '#bcbd22', '#17becf']

# Création du graphique à barres avec les couleurs personnalisées
fig, ax = plt.subplots(figsize=(10, 6))
bars = ax.barh(pays, proportions, color=couleurs, edgecolor='black', linewidth=1)

# Personnalisation du graphique
plt.xlabel('Proportion de sous-alimentation')
plt.ylabel('Pays')
plt.title('Les 10 pays avec la proportion la plus élevée de personnes sous-alimentées en 2017')

# Affichage des valeurs numériques au-dessus des barres
for i, bar in enumerate(bars):
    ax.annotate(f'{proportions[i]:.0%}', xy=(bar.get_width() + 0.005, bar.get_y() + bar.get_height() / 2),
                xytext=(5, 0), textcoords='offset points', va='center', fontsize=10, fontweight='bold', color='black')

# Inverser l'ordre des pays pour avoir le plus bas en haut
plt.gca().invert_yaxis()

# Ajustement des marges autour du graphique
plt.subplots_adjust(left=0.25)

# Ajout d'une grille
ax.grid(axis='x', linestyle='--', alpha=0.5)

# Amélioration de la mise en page
plt.tight_layout()

# Affichage du graphique
plt.show()
```



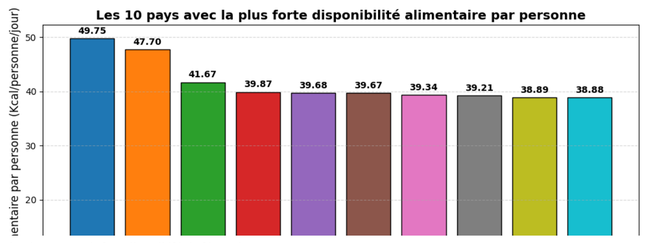

```
# Données des 10 pays qui ont bénéficié le plus de l'aide alimentaire depuis 2013 (en kg)
pays = ['République arabe syrienne', 'Éthiopie', 'Yémen', 'Soudan du Sud', 'Soudan', 'Kenya', 'Bangladesh', 'Somalie', 'République démocratique du Congo', 'Niger']
aide_alimentaire_kg = [1858943000, 1381294000, 1206484000, 695248000, 669784000, 552836000, 348188000, 292678000, 288502000, 276344000]

# Convertir l'aide alimentaire en tonnes (pour une meilleure lisibilité sur le graphique)
aide_alimentaire_tonnes = [aide / 1000000 for aide in aide_alimentaire_kg]

# Inverser l'ordre des pays et des valeurs pour obtenir un podium vertical
pays = pays[::-1]
aide_alimentaire_tonnes = aide_alimentaire_tonnes[::-1]

# Création du graphique en podium vertical (barres verticales) avec des couleurs personnalisées
plt.figure(figsize=(10, 6))
plt.barh(pays, aide_alimentaire_tonnes, color=['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd', '#8c564b', '#e377c2', '#7f7f7f', '#bcbd22', '#17becf'])

# Ajouter les valeurs au-dessus de chaque barre (podium vertical) avec un format lisible
for i, valeur in enumerate(aide_alimentaire_tonnes):
    plt.text(valeur, i, '{:,.0f}'.format(valeur) + "M tonnes", ha='left', va='center', fontsize=10, fontweight='bold')

# Personnalisation du graphique
plt.xlabel('Aide alimentaire (en millions de tonnes)')
plt.ylabel('Pays')
plt.title('Les 10 pays qui ont bénéficié le plus de l\'aide alimentaire depuis 2013', fontsize=14, fontweight='bold')

# Supprimer les bordures du graphique
plt.box(on=None)

# Affichage du graphique
plt.tight_layout()
plt.show()
```



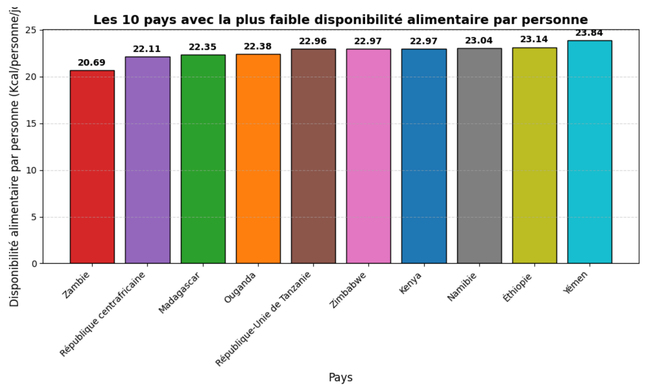

```
# Données de l'évolution de l'aide alimentaire pour chaque pays
annees_syrie = [2013, 2014, 2015, 2016]
aide_syrie = [563566000, 651870000, 524949000, 118558000]

annees_ethiopie = [2013, 2014, 2015]
aide_ethiopie = [591404000, 586624000, 203266000]

annees_yemen = [2013, 2014, 2015, 2016]
aide_yemen = [264764000, 103840000, 372306000, 465574000]

annees_soudan_sud = [2013, 2014, 2015]
aide_soudan_sud = [196330000, 450610000, 48308000]

annees_soudan = [2013, 2014, 2015]
aide_soudan = [330230000, 321904000, 17650000]

# Création des graphiques en ligne pour chaque pays
plt.plot(annees_syrie, aide_syrie, marker='o', label='République arabe syrienne')
plt.plot(annees_ethiopie, aide_ethiopie, marker='o', label='Éthiopie')
plt.plot(annees_yemen, aide_yemen, marker='o', label='Yémen')
plt.plot(annees_soudan_sud, aide_soudan_sud, marker='o', label='Soudan du Sud')
plt.plot(annees_soudan, aide_soudan, marker='o', label='Soudan')

# Personnalisation du graphique
plt.xlabel('Année')
plt.ylabel("Montant d'aide alimentaire")
plt.title("Évolution de l'aide alimentaire pour certains pays")
plt.legend()

# Ajustement des axes
plt.xticks(annees_syrie + annees_ethiopie + annees_yemen + annees_soudan_sud + annees_soudan)

# Amélioration de la mise en page
plt.grid(True)
plt.tight_layout()

# Affichage du graphique
plt.show()
```




```
# Données des pays et de leur disponibilité alimentaire par personne
pays = ['Turkménistan', 'Lesotho', 'Kiribati', 'Turquie', 'Autriche', 'Israël', 'Belgique', 'Monténégro', 'Samoa', 'Koweït']
disponibilite_alimentaire = [49.75, 47.70, 41.67, 39.87, 39.68, 39.67, 39.34, 39.21, 38.89, 38.88]

# Couleurs personnalisées pour les barres
couleurs = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd', '#8c564b', '#e377c2', '#7f7f7f', '#bcbd22', '#17becf']

# Création du graphique à barres avec les couleurs personnalisées
fig, ax = plt.subplots(figsize=(10, 6))
bars = ax.bar(pays, disponibilite_alimentaire, color=couleurs, edgecolor='black', linewidth=1)

# Personnalisation du graphique
plt.xlabel('Pays', fontsize=12)
plt.ylabel('Disponibilité alimentaire par personne (Kcal/personne/jour)', fontsize=12)
plt.title('Les 10 pays avec la plus forte disponibilité alimentaire par personne', fontsize=14, fontweight='bold')

# Ajout des valeurs numériques au-dessus des barres
for i, bar in enumerate(bars):
    height = bar.get_height()
    ax.annotate(f'{disponibilite_alimentaire[i]:.2f}', xy=(bar.get_x() + bar.get_width() / 2, height),
                xytext=(0, 3), textcoords='offset points', ha='center', va='bottom', fontsize=10, fontweight='bold', color='black')

# Rotation des étiquettes d'axe x pour une meilleure lisibilité
plt.xticks(rotation=45, ha='right', fontsize=10)

# Ajustement des marges autour du graphique
plt.subplots_adjust(bottom=0.2)

# Ajout d'une grille
ax.grid(axis='y', linestyle='--', alpha=0.5)

# Amélioration de la mise en page
plt.tight_layout()

# Affichage du graphique
plt.show()
```

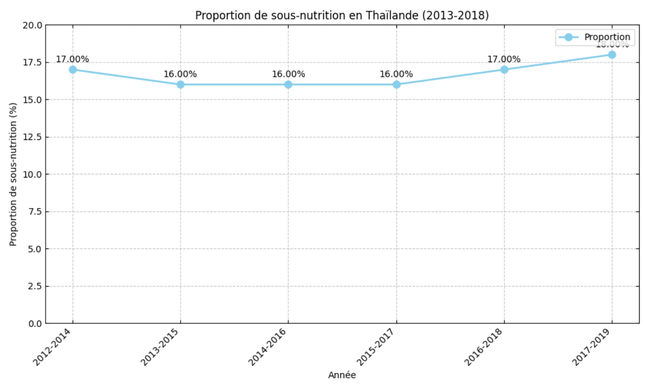



```
# Données des pays et de leur disponibilité alimentaire par personne
pays = ['Zambie', 'République centrafricaine', 'Madagascar', 'Ouganda', 'République-Unie de Tanzanie', 'Zimbabwe', 'Kenya', 'Namibie', 'Éthiopie', 'Yémen']
disponibilite_alimentaire = [20.69, 22.11, 22.35, 22.38, 22.96, 22.97, 22.97, 23.04, 23.14, 23.84]

# Couleurs personnalisées pour les barres
couleurs = ['#d62728', '#9467bd', '#2ca02c', '#ff7f0e', '#8c564b', '#e377c2', '#1f77b4', '#7f7f7f', '#bcbd22', '#17becf']

# Création du graphique à barres avec les couleurs personnalisées
fig, ax = plt.subplots(figsize=(10, 6))
bars = ax.bar(pays, disponibilite_alimentaire, color=couleurs, edgecolor='black', linewidth=1)

# Personnalisation du graphique
plt.xlabel('Pays', fontsize=12)
plt.ylabel('Disponibilité alimentaire par personne (Kcal/personne/jour)', fontsize=12)
plt.title('Les 10 pays avec la plus faible disponibilité alimentaire par personne', fontsize=14, fontweight='bold')

# Ajout des valeurs numériques au-dessus des barres
for i, bar in enumerate(bars):
    height = bar.get_height()
    ax.annotate(f'{disponibilite_alimentaire[i]:.2f}', xy=(bar.get_x() + bar.get_width() / 2, height),
                xytext=(0, 3), textcoords='offset points', ha='center', va='bottom', fontsize=10, fontweight='bold', color='black')

# Rotation des étiquettes d'axe x pour une meilleure lisibilité
plt.xticks(rotation=45, ha='right', fontsize=10)

# Ajustement des marges autour du graphique
plt.subplots_adjust(bottom=0.2)

# Ajout d'une grille
ax.grid(axis='y', linestyle='--', alpha=0.5)

# Amélioration de la mise en page
plt.tight_layout()

# Affichage du graphique
plt.show()
```

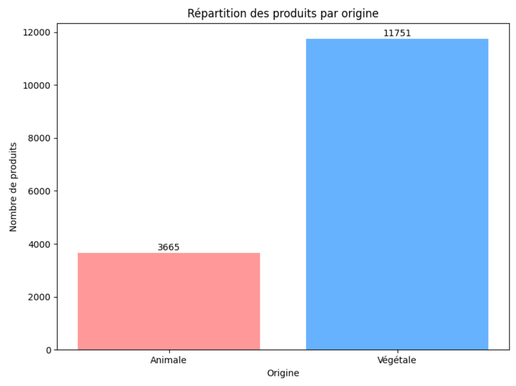


Graphiques des analyses complémentaires

```
# Données
annees = ['2012-2014', '2013-2015', '2014-2016', '2015-2017', '2016-2018', '2017-2019']
proportions = [0.17, 0.16, 0.16, 0.16, 0.17, 0.18]

# Conversion en pourcentage
proportions_percentage = [p * 100 for p in proportions]

# Création du graphique en ligne avec des améliorations visuelles
plt.figure(figsize=(10, 6))
plt.plot(annees, proportions_percentage, marker='o', color='skyblue', linewidth=2, markersize=8, label='Proportion')
plt.xlabel('Année')
plt.ylabel('Proportion de sous-nutrition (%)')
plt.title('Proportion de sous-nutrition en Thaïlande (2013-2018)')
plt.grid(True, linestyle='--', alpha=0.7)
plt.ylim(0, 20)  # Ajuster la plage de l'axe y pour une meilleure visualisation des proportions en pourcentage

# Ajout de légendes et de valeurs sur les points
for i, proportion in enumerate(proportions_percentage):
    plt.text(annees[i], proportion + 0.5, f'{proportion:.2f}%', ha='center', fontsize=10)

# Amélioration de l'apparence des axes
plt.tick_params(axis='both', which='both', direction='in', bottom=True, top=True, left=True, right=True)
plt.xticks(rotation=45, ha='right')
plt.legend()

plt.tight_layout()

# Affichage du graphique
plt.show()
```




```
# Données
origines = ['Animale', 'Végétale']
nombre_produits = [3665, 11751]

# Création du graphique à barres
plt.figure(figsize=(8, 6))
plt.bar(origines, nombre_produits, color=['#FF9999', '#66B2FF'])
plt.xlabel('Origine')
plt.ylabel('Nombre de produits')
plt.title('Répartition des produits par origine')

# Affichage des valeurs au-dessus des barres
for i, nombre in enumerate(nombre_produits):
    plt.text(i, nombre + 100, str(nombre), ha='center', fontsize=10)

plt.tight_layout()

# Affichage du graphique à barres
plt.show()
```

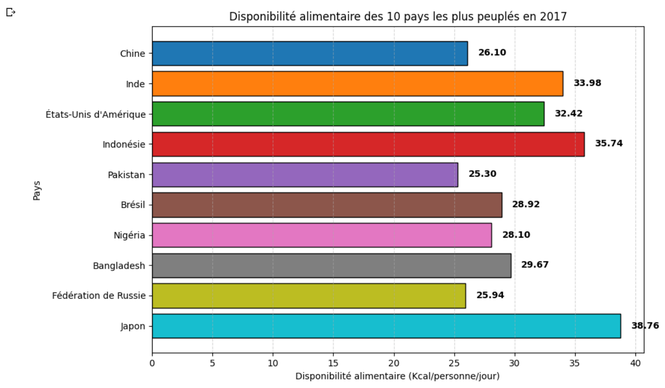



```
# Données des pays et des moyennes de disponibilité en kcal
pays = ['Ukraine', 'Venezuela', 'Argentine', 'Lesotho', 'Géorgie', 'Gambie', 'Mexique', 'Mauritanie', 'Nigéria', 'Mozambique']
moyennes_disponibilite_kcal = [34.69, 30.13, 31.89, 47.70, 31.94, 32.29, 25.64, 29.67, 34.48, 27.43]

# Couleurs personnalisées pour les barres
couleurs = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd', '#8c564b', '#e377c2', '#7f7f7f', '#bcbd22', '#17becf']

# Création du graphique à barres avec les couleurs personnalisées
fig, ax = plt.subplots(figsize=(10, 6))
bars = ax.barh(pays, moyennes_disponibilite_kcal, color=couleurs, edgecolor='black', linewidth=1)

# Personnalisation du graphique
plt.xlabel('Moyenne de disponibilité alimentaire (kcal)')
plt.ylabel('Pays')
plt.title('Moyenne de disponibilité alimentaire en kcal des 10 pays les plus sous-alimentés en 2017')

# Affichage des valeurs numériques au-dessus des barres
for i, bar in enumerate(bars):
    ax.annotate(f'{moyennes_disponibilite_kcal[i]:.2f} kcal', xy=(bar.get_width() + 0.5, bar.get_y() + bar.get_height() / 2),
                xytext=(5, 0), textcoords='offset points', va='center', fontsize=10, fontweight='bold', color='black')

# Inverser l'ordre des pays pour avoir le plus bas en haut
plt.gca().invert_yaxis()

# Ajustement des marges autour du graphique
plt.subplots_adjust(left=0.25)

# Ajout d'une grille
ax.grid(axis='x', linestyle='--', alpha=0.5)

# Amélioration de la mise en page
plt.tight_layout()

# Affichage du graphique
plt.show()
```



```
# Données des pays et de la disponibilité alimentaire en kcal/personne/jour
pays = ['Chine','Inde','États-Unis d\'Amérique','Indonésie','Pakistan','Brésil','Nigéria','Bangladesh', 'Fédération de Russie', 'Japon']
disponibilite_alimentaire = [26.10, 33.98, 32.42, 35.74, 25.30, 28.92, 28.10, 29.67, 25.94, 38.76]

# Couleurs personnalisées pour les barres
couleurs = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd', '#8c564b', '#e377c2', '#7f7f7f', '#bcbd22', '#17becf']

# Création du graphique à barres avec les couleurs personnalisées
fig, ax = plt.subplots(figsize=(10, 6))
bars = ax.barh(pays, disponibilite_alimentaire, color=couleurs, edgecolor='black', linewidth=1)

# Personnalisation du graphique
```

```
plt.xlabel('Disponibilité alimentaire (Kcal/personne/jour)')
plt.ylabel('Pays')
plt.title('Disponibilité alimentaire des 10 pays les plus peuplés en 2017')

# Affichage des valeurs numériques au-dessus des barres
for i, bar in enumerate(bars):
    ax.annotate(f'{disponibilite_alimentaire[i]:.2f}', xy=(bar.get_width() + 0.5, bar.get_y() + bar.get_height() / 2),
                xytext=(5, 0), textcoords='offset points', va='center', fontsize=10, fontweight='bold', color='black')

# Inverser l'ordre des pays pour avoir le plus bas en haut
plt.gca().invert_yaxis()

# Ajustement des marges autour du graphique
plt.subplots_adjust(left=0.25)

# Ajout d'une grille
ax.grid(axis='x', linestyle='--', alpha=0.5)

# Amélioration de la mise en page
plt.tight_layout()

# Affichage du graphique
plt.show()
```



0 s terminée à 21:29