```python
# -*- coding: utf-8 -*-
from pathlib import Path
import csv
from lxml import etree

def basic(row):
    """Prend en entrée une ligne de CSV sous forme de liste et crée les différents éléments de la modélisation XML pour les informations basiques"""

    #on  définit la variable game comme global pour qu'elle soit
    #accessible par les autres fonctions
    global game

    #élément game inclus dans la racine
    #un attribut rank qui vient du champ "rank"
    game = etree.SubElement(root,"game")
    game.set("rank",row[0])

    #élément name inclus dans game
    #éléments official et ascii inclus dans name : valeurs des champs "Name" et "basename"
    name = etree.SubElement(game,"name")
    official = etree.SubElement(name,"official")
    official.text = row[1]
    infoascii = etree.SubElement(name,"ascii")
    infoascii.text = row[2]

    #élément genre inclus game
    #valeur du champ "Genre"
    genre = etree.SubElement(game,"genre")
    genre.text = row[3]

    #élément platform inclus dans game
    #valeur du champ "Platform"
    platform = etree.SubElement(game,"platform")
    platform.text = row[4]

    #élément production inclus dans game
    #éléments publisher et developer inclus dans production : valeurs des champs "Publisher" et "Developer"
    production = etree.SubElement(game,"production")
    publisher = etree.SubElement(production,"publisher")
    publisher.text = row[5]
    developer = etree.SubElement(production,"developer")
    developer.text = row[6]

    #élément year inclus dans game
    #valeur du champ "Year"
    year = etree.SubElement(game,"year")
    year.text = row[7]

def sales(row):
    """Prend en entrée une ligne de CSV sous forme de liste et crée les différents éléments de la modélisation XML pour les informations de ventes"""

    #élément distribution inclus dans game
    distribution = etree.SubElement(game,"distribution")

    #élément sales inclus dans distribution
    #son attribut area aura comme valeur "global" et contenu de l'élément = champ "Global_Sales"
    sales1 = etree.SubElement(distribution,"sales")
    sales1.set("area","global")
    sales1.text = row[8]

    #élément sales inclus dans distribution
    #son attribut area aura comme valeur "usa" et contenu de l'élément = champ "NA_Sales"
    sales2 = etree.SubElement(distribution,"sales")
    sales2.set("area","usa")
    sales2.text = row[9]

```

```
67          #élément sales inclus dans distribution
68          #son attribut area aura comme valeur "europe" et contenu de l'élément = champ
            "PAL_Sales"
69          sales3 = etree.SubElement(distribution,"sales")
70          sales3.set("area","europe")
71          sales3.text = row[10]
72
73          #élément sales inclus dans distribution
74          #son attribut area aura comme valeur "japan" et contenu de l'élément = champ
            "JP_Sales"
75          sales4 = etree.SubElement(distribution,"sales")
76          sales4.set("area","japan")
77          sales4.text = row[11]
78
79      def critic(row):
80          """Prend en entrée une ligne de CSV sous forme de liste et crée l'élément
81          de la modélisation XML pour la note du jeu obtenue de la critique"""
82
83          #élément critic_score inclus dans game
84          #valeur du champ "Critic_Score"
85          critic = etree.SubElement(game,"critic_score")
86          critic.text = row[12]
87
88      def esrb(row):
89          """Prend en entrée une ligne de CSV sous forme de liste et crée l'élément
90          de la modélisation XML pour le symbole ESRB du jeu"""
91
92          #élément esrb_rating inclus dans game
93          #valeur du champ "ESRB_Rating"
94          esrb = etree.SubElement(game,"esrb_rating")
95          esrb.text = row[12]
96
97      #répertoire contenant les fichiers CSV
98      directory = Path("../data/CSV")
99
100     #répertoire de sortie qui contiendra les fichiers XML
101     outDir = Path("../xml")
102
103     #itération dans le répertoire des CSV pour lire chaque jeu de données avec le reader
        CSV
104     #on vérifie qu'on ait bien un fichier
105     #et qu'il ne s'agit pas du jeu de données CSV d'origine ("vgsales-12-4-2019.csv")
106     for child in directory.iterdir():
107         if child.is_file():
108             if child.match("vgsales*"):
109                 continue
110             with open(child,"r",encoding="utf-8") as file:
111
112                 print(f'Reading "{child.name}" to generate an XML document...\n')
113
114                 #on initialise une variable qui contiendra une lettre pour savoir quel
                    jeu de données on a
115                 nameF = ""
116
117                 #on récupére le nom du fichier sans l'extension et on ajoute l'extension
                    ".xml"
118                 #pour avoir le nom de fichier de sortie
119                 nameOut = child.stem + ".xml"
120
121                 #on crée le chemin pour accéder à ce fichier de sortie
122                 pathToF = outDir.joinpath(nameOut)
123
124                 #objet reader pour lire proprement le fichier
125                 table = csv.reader(file,delimiter = ",")
126
127                 #on passe une ligne, celle des en-têtes
128                 next(table)
129
130                 #racine collection de chaque fichier XML
131                 root = etree.Element("collection")
132
133                 #on va parcourir les lignes du jeu de données
```

```
134             #pour agrandir l'arborescence XML (etree) au fur et à mesure pour chaque
                modélisation
135             #on fait des tests pour vérifier quel jeu de données on a et on donne
                une lettre à notre variable de nom
136             #on appelle ensuite les bonnes fonctions sur la ligne pour agrandir
                notre arbre
137             for row in table:
138                 if child.match("basic.csv"):
139                     nameF = "B"
140                     basic(row)
141                 if child.match("plus_sales.csv"):
142                     nameF = "S"
143                     basic(row)
144                     sales(row)
145                 if child.match("plus_critic*.csv"):
146                     nameF = "C"
147                     basic(row)
148                     sales(row)
149                     critic(row)
150                 if child.match("plus_esrb*.csv"):
151                     nameF = "E"
152                     basic(row)
153                     sales(row)
154                     esrb(row)
155
156             #quand une lecture de jeu de données est finie
157             #on fait des tests pour savoir quel jeu de données on a
158             #on récupère l'arbre XML que les fonctions ont agrandi au fur et à mesure
159             #et on l'écrit dans le fichier correspondant
160             #en ajoutant une déclaration XML ainsi qu'une indentation
161             if nameF == "B":
162                 tree = etree.ElementTree(root)
163                 tree.write(str(pathToF), encoding="utf-8",xml_declaration=True,
                    method="xml",pretty_print=True)
164                 print(f'XML document "{nameOut}" generated ! \n')
165                 print("------------------------------")
166
167             if nameF == "S":
168                 tree = etree.ElementTree(root)
169                 tree.write(str(pathToF), encoding="utf-8",xml_declaration=True,
                    method="xml",pretty_print=True)
170                 print(f'XML document "{nameOut}" generated ! \n')
171                 print("------------------------------")
172
173             if nameF == "C":
174                 tree = etree.ElementTree(root)
175                 tree.write(str(pathToF), encoding="utf-8",xml_declaration=True,
                    method="xml",pretty_print=True)
176                 print(f'XML document "{nameOut}" generated ! \n')
177                 print("------------------------------")
178
179             if nameF == "E":
180                 tree = etree.ElementTree(root)
181                 tree.write(str(pathToF), encoding="utf-8",xml_declaration=True,
                    method="xml",pretty_print=True)
182                 print(f'XML document "{nameOut}" generated ! \n')
183                 print("------------------------------")
```