```
import datetime


class Date:
    def __init__(self, jour, mois, annee):
        self.jour = int(jour)
        self.mois = int(mois)
        self.annee = int(annee)

    def __str__(self):
        return f"{self.annee}-{self.jour:02d}-{self.mois:02d}"

    def est_passee(self):
        """Renvoie True si la date est passée, False sinon."""
        aujourd_hui = datetime.date.today()
        date_event = datetime.date(self.annee, self.mois, self.jour)
        return date_event < aujourd_hui


class Evenement:
    def __init__(self, nom_evenement, place_evenement, date_evenement, contact_evenement,
prix_evenement):
        self.nom = nom_evenement
        self.place = place_evenement
        self.date = date_evenement
        self.contact = contact_evenement
        self.prix = prix_evenement

    def nommer (self):
        print(f"le nom de l'evenement que vous avez choisis est {self.nom}")

    def place (self):
        print(f"la place de l'evenement que vous avez choisis est {self.place}")

    def date (self) :
        print(f" la date que vous avez choisis pour votre evenement est {self.date}")

    def contact (self) :
        print(f" le contact que vous avez choisis pour votre evenement est
{self.contact}")

    def prix (self) :
        print(f" le prix que vous avez déterminé pour votre evenement est {self.prix}")

    def __str__(self):
        return f"{self.nom} au {self.place} le {self.date}. GRATUIT = {self.prix}.
CONTACT: {self.contact}"


# Fonction créée par Julien
def CreerID(dataDict):
    return len(dataDict)


def CreerEvenement(dataDict, nom_evenement: str, place_evenement:str, date_evenement:
Date, contact_evenement: str, prix_evenement: bool):
    id = CreerID(dataDict)
    dataDict[id] = Evenement(nom_evenement, place_evenement, date_evenement,
contact_evenement, prix_evenement)
```