Algorithm 1 Algorithme Glouton

Require: *df_ville*, *df_object*, *capacite*

Ensure: Liste des villes visitées π, objets pris *obj_pris*, poids total *poids_tot*, dictionnaire des objets pris par ville *dict_ville_objet_pris*

poids_tot $\leftarrow 0$
nb_objet $\leftarrow$ longueur des objets dans *df_object*
obj_pris $\leftarrow$ tableau de zéros de taille *nb_objet*
nb_ville $\leftarrow$ longueur des villes dans *df_ville*
critere_limite $\leftarrow$ *capacite*/*nb_ville*
villes_dispo $\leftarrow$ liste des indices des villes dans *df_ville*
ville_actuelle $\leftarrow 1$
Retirer *ville_actuelle* de *villes_dispo*
Initialiser les dictionnaires *dict_ville_objet* et *dict_ville_objet_pris*
while *capacite* $>$ *poids_tot* **and** *villes_dispo* $\neq \emptyset$ **do**
    Calculer *distance_table* $\leftarrow$ distances depuis *ville_actuelle*
    ville_cible_best $\leftarrow$ ville dans *villes_dispo* maximisant l'évaluation (critère glouton)
    Trier les objets disponibles dans *ville_cible_best* par rapport au rapport $\frac{\text{Profit}}{\text{Poids}}$
    for chaque objet *obj* dans la liste triée **do**
        if *poids_obj* $\leq$ *capacite* $-$ *poids_tot* **then**
            Ajouter *obj* à *dict_ville_objet_pris*[*ville_cible_best*]
            poids_tot $\leftarrow$ *poids_tot* $+$ *poids_obj*
            Mettre à jour *obj_pris*
        end if
    end for
    Mettre à jour π et *ville_actuelle*
end while
Retourner π, *obj_pris*, *poids_tot*, *dict_ville_objet_pris*