■ Simulation de Pompe à Essence

DM Python · BTS SIO SISR · Christophe Beaudenon · E2SE Digital IT Caen

---

PARTIE 1 — Sujet du TP

Contexte

Créer un programme qui simule le fonctionnement d'une pompe à essence. Le programme permet à l'utilisateur de choisir un type de carburant, de spécifier la quantité ou le montant à payer, et de recevoir un récapitulatif de son achat. Le programme doit également gérer les stocks de carburant.

Initialisation

- Trois types de carburant : Sans Plomb 95, Sans Plomb 98, Diesel
- Quantités de départ : 500 litres par carburant
- Prix par litre : SP95 → 1,70 €/L · SP98 → 1,80 €/L · Diesel → 1,60 €/L

Fonctionnalités obligatoires

■ Faire un plein

– Sélectionner un type de carburant

– Insérer sa carte et saisir son code PIN (3 essais)

– Entrer le nombre de litres ou le montant en euros

– Vérifier le stock disponible

– Afficher la quantité et le montant · Déduire du stock

■ Afficher les stocks

– Afficher les quantités restantes de chaque carburant

■ Quitter

– Message de sortie et fin propre du programme

Bonus (facultatif — implémentés)

✓ Alerte stock bas : avertissement si le stock passe sous 50 L

✓ Rechargement de la pompe : option admin (code 9999) pour réinitialiser un stock

✓ Suivi des transactions : historique de chaque plein (carburant, litres, montant)

PARTIE 2 — Code source Python

■ *Beaudenon_DM_python_Essence.py*

```
# Simulation de pompe à essence

CARBURANTS = {
"1": {"nom": "SP95", "prix": 1.70, "stock": 500.0, "stock_initial": 500.0},
"2": {"nom": "SP98", "prix": 1.80, "stock": 500.0, "stock_initial": 500.0},
"3": {"nom": "Diesel", "prix": 1.60, "stock": 500.0, "stock_initial": 500.0},
}

SEUIL_STOCK_BAS = 50.0
CODE_PIN_CORRECT = "1234"
CODE_ADMIN = "9999" # code admin pour la recharge des stocks

historique = [] # liste de dictionnaires : carburant, litres, montant
```

```
# affichage menu #

def afficher_menu():
print("\n=== POMPE À ESSENCE ===")
print("1. Faire un plein")
print("2. Afficher les stocks")
print("3. Recharger les stocks")
print("4. Afficher l'historique")
print("0. Quitter")


def choisir_carburant():
print("\nChoix du carburant :")
for key, data in CARBURANTS.items():
print(f"{key}. {data['nom']} - {data['prix']} €/L - Stock : {data['stock']:.2f} L")
choix = input("Votre choix : ").strip()
if choix in CARBURANTS:
return choix
print("Choix invalide.")
return None


def verifier_code_pin():
essais_restants = 3
while essais_restants > 0:
code = input(f"Entrez votre code PIN (essais restants : {essais_restants}) : ").strip()
if code == CODE_PIN_CORRECT:
print("Code accepté.")
return True
else:
print("Code incorrect.")
essais_restants -= 1
print("Trop d'essais. Paiement refusé.")
return False


def verifier_code_admin():
essais_restants = 3
while essais_restants > 0:
code = input(f"Code ADMIN requis (essais restants : {essais_restants}) : ").strip()
if code == CODE_ADMIN:
print("Accès admin autorisé.")
return True
else:
print("Code admin incorrect.")
essais_restants -= 1
print("Accès admin refusé.")
return False


def faire_plein():
carburant_id = choisir_carburant()
if carburant_id is None:
return

carburant = CARBURANTS[carburant_id]

# Simulation carte bancaire
if not verifier_code_pin():
return

print("\nSouhaitez-vous saisir :")
print("1. Un nombre de litres")
print("2. Un montant en euros")
choix = input("Votre choix : ").strip()
```

```
litres = 0.0
montant = 0.0

if choix == "1":
try:
litres = float(input("Nombre de litres souhaités : ").strip())
if litres <= 0:
print("Quantité invalide.")
return
montant = litres * carburant["prix"]
except ValueError:
print("Saisie invalide.")
return
elif choix == "2":
try:
montant = float(input("Montant en euros : ").strip())
if montant <= 0:
print("Montant invalide.")
return
litres = montant / carburant["prix"]
except ValueError:
print("Saisie invalide.")
return
else:
print("Choix invalide.")
return

# Vérification du stock
if litres > carburant["stock"]:
print("Stock insuffisant pour cette quantité.")
print(f"Stock disponible : {carburant['stock']:.2f} L")
return

# Réalisation du plein
carburant["stock"] -= litres

print("\n=== RÉCAPITULATIF ===")
print(f"Carburant : {carburant['nom']}")
print(f"Quantité : {litres:.2f} L")
print(f"Montant : {montant:.2f} €")

# Historique
historique.append(
{
"carburant": carburant["nom"],
"litres": litres,
"montant": montant,
}
)

# Alerte stock bas
if carburant["stock"] < SEUIL_STOCK_BAS:
print(f"ATTENTION : stock bas pour {carburant['nom']} ({carburant['stock']:.2f} L).")


def afficher_stocks():
print("\n=== STOCKS DE CARBURANT ===")
for data in CARBURANTS.values():
ligne = f"{data['nom']} : {data['stock']:.2f} L"
if data["stock"] < SEUIL_STOCK_BAS:
ligne += " (STOCK BAS)"
print(ligne)
```

```
def recharger_stocks():
print("\n=== RECHARGER LES STOCKS ===")

# Vérification du code admin
if not verifier_code_admin():
return

print("Vous allez réinitialiser le stock d'un carburant à sa valeur de départ.")
carburant_id = choisir_carburant()
if carburant_id is None:
return

carburant = CARBURANTS[carburant_id]

confirmation = input(
f"Confirmez la réinitialisation du stock de {carburant['nom']} "
f"à {carburant['stock_initial']:.2f} L ? (o/n) : "
).strip().lower()

if confirmation != "o":
print("Opération annulée.")
return

carburant["stock"] = carburant["stock_initial"]
print(
f"Le stock de {carburant['nom']} a été réinitialisé à "
f"{carburant['stock']:.2f} L."
)


def afficher_historique():
print("\n=== HISTORIQUE DES TRANSACTIONS ===")
if not historique:
print("Aucune transaction pour le moment.")
return
for i, t in enumerate(historique, start=1):
print(
f"{i}. Carburant : {t['carburant']}, "
f"Quantité : {t['litres']:.2f} L, "
f"Montant : {t['montant']:.2f} €"
)


def main():
while True:
afficher_menu()
choix = input("Votre choix : ").strip()

if choix == "1":
faire_plein()
elif choix == "2":
afficher_stocks()
elif choix == "3":
recharger_stocks()
elif choix == "4":
afficher_historique()
elif choix == "0":
print("Merci, au revoir.")
break
else:
print("Choix invalide, veuillez réessayer.")


if __name__ == "__main__":
main()
```

PARTIE 3 — Description des fonctions

Fonction	Rôle
`afficher_menu()`	Affiche le menu principal avec les 5 options disponibles.
`choisir_carburant()`	Affiche la liste des carburants avec prix et stock. Retourne l'identifiant choisi ou None si invalide.
`verifier_code_pin()`	Simule la validation carte bancaire. 3 essais maximum. Retourne True si accepté, False si échec.
`verifier_code_admin()`	Vérifie le code admin (9999) pour l'accès à la recharge. 3 essais maximum.
`faire_plein()`	Fonction principale : sélection carburant → PIN → choix litres/€ → vérification stock → débit → récapitulatif → historique → alerte si stock bas.
`afficher_stocks()`	Affiche le stock actuel de chaque carburant. Signale STOCK BAS si < 50 L.
`recharger_stocks()`	Après vérification admin, réinitialise le stock d'un carburant à sa valeur initiale (500 L).
`afficher_historique()`	Liste toutes les transactions enregistrées (carburant, litres, montant).
`main()`	Boucle principale du programme. Lit le choix utilisateur et appelle la fonction correspondante.