

# Compte rendu du TP N° 5

## Introduction

Ce travail pratique consiste à concevoir et simuler une mini-plateforme de vente en ligne appelée ShopExpress++.

L'application permet de gérer différents types de produits (électroniques, alimentaires et vêtements) à l'aide des concepts de la programmation orientée objet en C++.

Le TP met en œuvre l'héritage, le polymorphisme et la gestion d'une collection de produits dans un catalogue.

## Objectif du TP

L'objectif principal de ce TP est de :

- Comprendre et appliquer les principes de la programmation orientée objet ;
- Utiliser l'héritage et le polymorphisme pour gérer plusieurs types de produits ;
- Manipuler des pointeurs et des collections polymorphes ;
- Simuler le fonctionnement d'un catalogue de vente et d'un service de statistiques.

```

h Produit.h > ...
1  #ifndef PRODUIT_H
2  #define PRODUIT_H
3  #include<string>
4  #include<ostream>
5
6  using namespace std;
7  class ServiceStatistique; // déclaration anticipée
8
9  class Produit {
10 protected:
11     int id;
12     string nom;
13     double prixHT;
14     int stock;
15
16 public:
17     // constructeur
18     Produit(int id, string nom, double prixHT, int stock);
19
20     // getters
21     int getStock() const;
22     double getPrixHT() const;
23
24     // méthodes virtuelles
25     virtual void afficher();
26     virtual double prixTTC();
27
28     // destructeur virtuel
29     virtual ~Produit();
30
31     // classe amie
32     friend class ServiceStatistique;
33 };
34
35 #endif // PRODUIT_H

```

```

C++ Produit.cpp > ...
1  #include "Produit.h"
2  #include <iostream>
3
4  Produit::Produit(int id, string nom, double prixHT, int stock)
5      : id(id), nom(nom), prixHT(prixHT), stock(stock) {}
6
7  int Produit::getStock() const {
8      return stock;
9  }
10
11 double Produit::getPrixHT() const {
12     return prixHT;
13 }
14
15 void Produit::afficher() {
16     cout << "ID: " << id
17         << " | Nom: " << nom
18         << " | Prix HT: " << prixHT
19         << " | Stock: " << stock << endl;
20 }
21
22 double Produit::prixTTC() {
23     return prixHT * 1.20; // TVA 20%
24 }
25
26 Produit::~Produit() {}
27

```

```

h ProduitAlimentaire.h > ...
1  #ifndef PRODUITALIMENTAIRE_H
2  #define PRODUITALIMENTAIRE_H
3
4  #include "Produit.h"
5
6  class ProduitAlimentaire : public Produit {
7  private:
8      string datePeremption;
9      string origine;
10
11  public:
12      ProduitAlimentaire(int id, string nom, double prixHT, int stock,
13                          string datePeremption, string origine);
14
15      void afficher() override;
16      double prixTTC() override;
17  };
18
19  #endif
20

```

```

C++ ProduitAlimentaire.cpp > ...
1  #include "ProduitAlimentaire.h"
2  #include <iostream>
3  using namespace std;
4
5  ProduitAlimentaire::ProduitAlimentaire(int id, string nom, double prixHT, int stock,
6                                          string datePeremption, string origine)
7      : Produit(id, nom, prixHT, stock),
8        datePeremption(datePeremption), origine(origine) {}
9
10 void ProduitAlimentaire::afficher() {
11     Produit::afficher();
12     cout << "Date peremption: " << datePeremption
13          << " | Origine: " << origine << endl;
14 }
15
16 double ProduitAlimentaire::prixTTC() {
17     return prixHT * 1.07; // TVA 7%
18 }
19

```

```

h produitElectronique.h > ...
1  #ifndef PRODUITELECTRONIQUE_H
2  #define PRODUITELECTRONIQUE_H
3
4  #include "Produit.h"
5
6  class ProduitElectronique : public Produit {
7  private:
8      int dureeGarantie;
9
10  public:
11      ProduitElectronique(int id, string nom, double prixHT, int stock, int dureeGarantie);
12
13      void afficher() override;
14      double prixTTC() override;
15  };
16
17  #endif
18

```

```

C++ produitElectronique.cpp > afficher()
1  #include "ProduitElectronique.h"
2  #include <iostream>
3  using namespace std;
4
5  ProduitElectronique::ProduitElectronique(int id, string nom, double prixHT, int stock, int dureeGarantie)
6  : Produit(id, nom, prixHT, stock), dureeGarantie(dureeGarantie) {}
7
8  void ProduitElectronique::afficher() {
9      Produit::afficher();
10     cout << "Garantie: " << dureeGarantie << " mois" << endl;
11 }
12
13 double ProduitElectronique::prixTTC() {
14     return prixHT * 1.20 + 10; // TVA + éco-tax
15 }
16

```

```

h ProduitVetement.h > ...
1  #ifndef PRODUITVETEMENT_H
2  #define PRODUITVETEMENT_H
3
4  #include "Produit.h"
5
6  class ProduitVetement : public Produit {
7  private:
8      string taille;
9      string genre;
10
11 public:
12     ProduitVetement(int id, string nom, double prixHT, int stock,
13                     string taille, string genre);
14
15     void afficher() override;
16 };
17
18 #endif
19

```

```

C++ ProduitVetement.cpp > ...
1  #include "ProduitVetement.h"
2  #include <iostream>
3  using namespace std;
4
5  ProduitVetement::ProduitVetement(int id, string nom, double prixHT, int stock,
6                                   string taille, string genre)
7  : Produit(id, nom, prixHT, stock),
8    taille(taille), genre(genre) {}
9
10 void ProduitVetement::afficher() {
11     Produit::afficher();
12     cout << "Taille: " << taille
13         << " | Genre: " << genre << endl;
14 }
15

```

```

h ServiceStatistique.h > ...
1  #ifndef SERVICESTATISTIQUE_H
2  #define SERVICESTATISTIQUE_H
3
4  #include <list>
5  #include "Produit.h"
6
7  class ServiceStatistique {
8  public:
9      double valeurTotaleStock(const list<Produit*>& catalogue);
10     Produit* produitLePlusStoque(const list<Produit*>& catalogue);
11 };
12
13 #endif
14

```

```

C+ ServiceStatistique.cpp > ...
1  #include "ServiceStatistique.h"
2
3  double ServiceStatistique::valeurTotaleStock(const list<Produit*>& catalogue) {
4      double total = 0;
5      for (Produit* p : catalogue) {
6          total += p->prixHT * p->stock;
7      }
8      return total;
9  }
10
11  Produit* ServiceStatistique::produitLePlusStoque(const list<Produit*>& catalogue) {
12      Produit* max = nullptr;
13      for (Produit* p : catalogue) {
14          if (!max || p->stock > max->stock)
15              max = p;
16      }
17      return max;
18  }
19

```

```

h Catalog.h > ...
1  #ifndef CATALOG_H
2  #define CATALOG_H
3
4  #include <list>
5  #include "Produit.h"
6
7  class Catalog {
8  private:
9      list<Produit*> catalogue;
10
11  public:
12      void ajouterProduit(Produit* p);
13      void afficherCatalogue();
14      Produit* produitLePlusCher();
15
16      const list<Produit*>& getCatalogue() const;
17  };
18
19 #endif

```

```
C++ Catalog.cpp > ...
1  ∨ #include "Catalog.h"
2  ∨ #include <iostream>
3  ∨ using namespace std;
4
5  ∨ void Catalog::ajouterProduit(Produit* p) {
6  |     catalogue.push_back(p);
7  | }
8
9  ∨ void Catalog::afficherCatalogue() {
10 ∨     for (Produit* p : catalogue) {
11 |         p->afficher();
12 |         cout << "Prix TTC: " << p->prixTTC() << endl << endl;
13 |     }
14 }
15
16 ∨ Produit* Catalog::produitLePlusCher() {
17 |     Produit* max = nullptr;
18 |     for (Produit* p : catalogue) {
19 |         if (!max || p->prixTTC() > max->prixTTC())
20 |             max = p;
21 |     }
22 |     return max;
23 }
24
25 ∨ const list<Produit*>& Catalog::getCatalogue() const {
26 |     return catalogue;
27 }
```

```

G+ main.cpp > ...
1  #include <iostream>
2  #include "ProduitElectronique.h"
3  #include "ProduitAlimentaire.h"
4  #include "ProduitVetement.h"
5  #include "Catalog.h"
6  #include "ServiceStatistique.h"
7
8  using namespace std;
9
10 int main() {
11     Catalog catalog;
12
13     Produit* p1 = new ProduitElectronique(1, "PC", 7000, 5, 24);
14     Produit* p2 = new ProduitElectronique(2, "Telephone", 4000, 10, 12);
15     Produit* p3 = new ProduitAlimentaire(3, "Lait", 2.5, 50, "01/01/2026", "Maroc");
16     Produit* p4 = new ProduitAlimentaire(4, "Fromage", 15, 20, "10/12/2025", "France");
17     Produit* p5 = new ProduitVetement(5, "Tshirt", 80, 30, "M", "Homme");
18
19     catalog.ajouterProduit(p1);
20     catalog.ajouterProduit(p2);
21     catalog.ajouterProduit(p3);
22     catalog.ajouterProduit(p4);
23     catalog.ajouterProduit(p5);
24
25     cout << "-----Catalogue-----" << endl;
26     catalog.afficherCatalogue();
27
28     cout << "-----Produit le plus cher-----" << endl;
29     catalog.produitLePlusCher()->afficher();
30
31     ServiceStatistique stats;
32     cout << "\nProduit le plus stocké :" << endl;
33     stats.produitLePlusStocke(catalog.getCatalogue())->afficher();
34
35     return 0;
36 }

```

```

>>
Catalogue
ID: 1 -- Nom: Tbalette -- Prix HT: 7000 -- Stock: 5
Garantie: 24 mois
Prix TTC: 8410

ID: 2 -- Nom: iphone -- Prix HT: 4000 -- Stock: 10
Garantie: 12 mois
Prix TTC: 4810

ID: 3 -- Nom: beurre -- Prix HT: 2.5 -- Stock: 50
Date peremption: 01/01/2027 | Origine: Maroc
Prix TTC: 2.675

ID: 4 -- Nom: Fromage -- Prix HT: 15 -- Stock: 20
Date peremption: 20/12/2025 | Origine: Algerie
Prix TTC: 16.05

ID: 5 -- Nom: hoodie -- Prix HT: 80 -- Stock: 30
Taille: F | Genre: Femme
Prix TTC: 96

Produit le plus cher
ID: 1 -- Nom: Tbalette -- Prix HT: 7000 -- Stock: 5
Garantie: 24 mois

Produit le plus stocké :
ID: 3 -- Nom: beurre -- Prix HT: 2.5 -- Stock: 50
Date peremption: 01/01/2027 | Origine: Maroc

```

## Conclusion

À travers ce TP, nous avons pu mettre en pratique les notions fondamentales de la programmation orientée objet en C++.

La simulation de la plateforme ShopExpress++ montre l'intérêt d'une conception orientée objet pour modéliser un système réel de vente en ligne.

Ce travail permet ainsi de mieux comprendre la structuration du code, la réutilisabilité des classes et l'importance du polymorphisme dans des applications complexes.