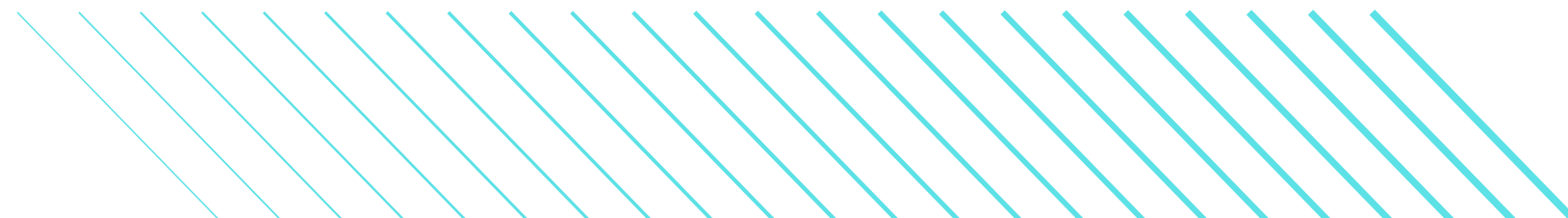
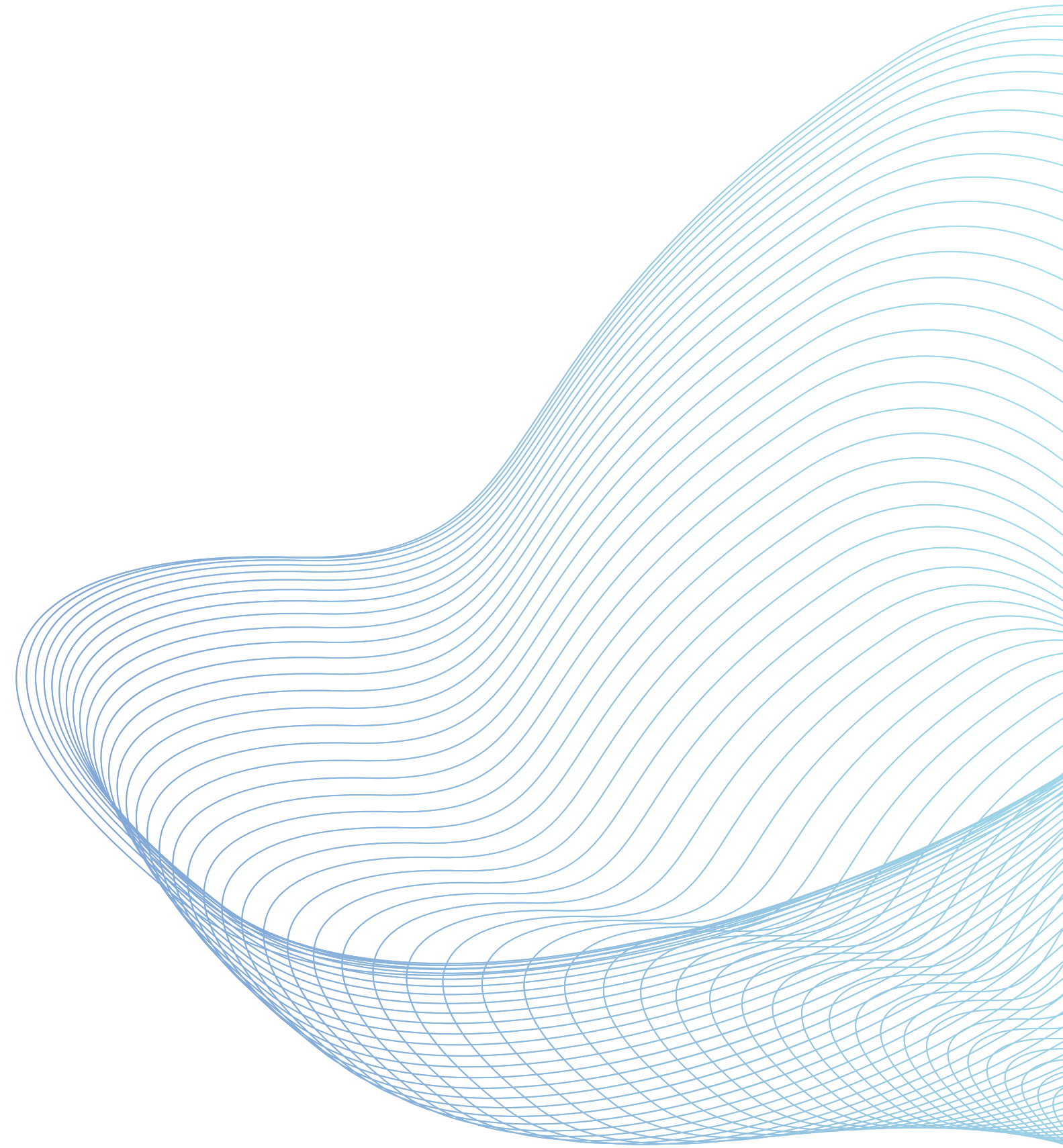


MINI PROJET

C++

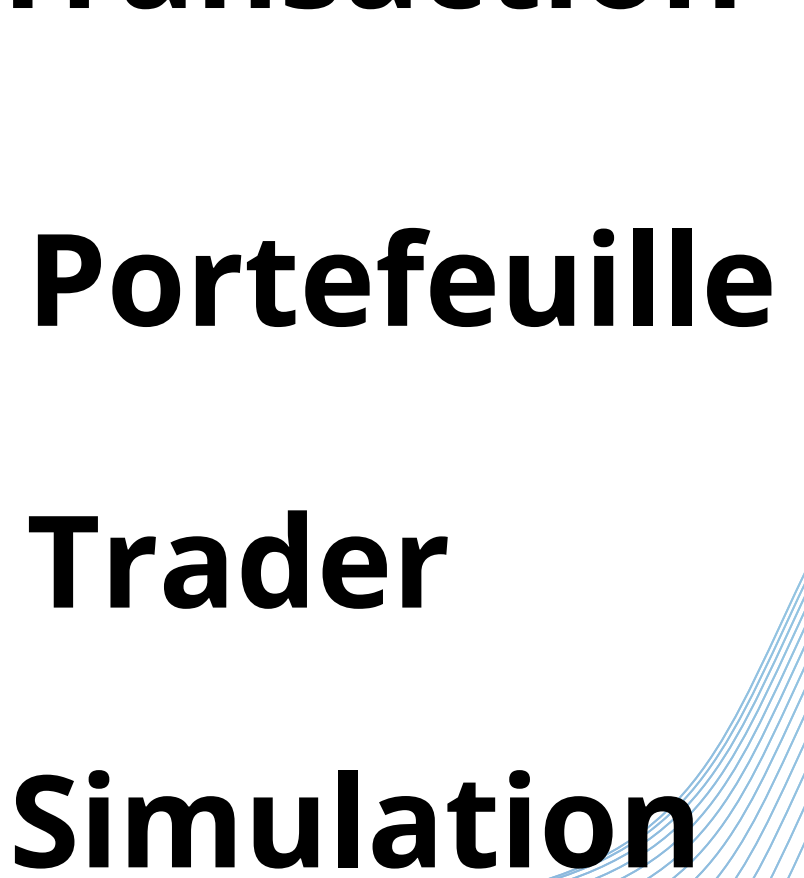
 **Sujet:**
Simulateur de trading



DESCRIPTION DU SUJET

L'objectif est de créer une application qui offre à l'utilisateur la possibilité de faire une simulation de trading en bourse

CONCEPTION ET DIAGRAMME DE CLASSES:

- 1 **Classe Date**
 - 2 **Classe Prix Journalier**
 - 3 **Classe Persistance Prix Journaliers**
 - 4 **Classe Bourse**
 - 5 **Classe Titre**
 - 6 **Classe Transaction**
 - 7 **Classe Portefeuille**
 - 8 **Classe Trader**
 - 9 **Classe Simulation**
- 

CLASSE BOURSE

Class fille BourseVector

Class fille BourseSet

Class fille BourseMultimap

CLASSE TRADER

Class fille Trader Aleatoire

Class fille Trader Algorithmique

METHODES DE LA CLASSE BOURSE :

getPrixJournaliersParDateEtPrix

```
vector<PrixJournalier> BourseVector::getPrixJournalierParDateEtPrix(double solde) const{  
    vector<PrixJournalier> vl={};  
  
    for(auto it:this->historique){  
        if(this->dateAujourd'hui <= it.getDate()){  
            if(this->dateAujourd'hui==it.getDate() && it.getPrix()<=solde){  
                vl.push_back(it);  
            }  
        }  
    }  
  
    return vl;  
}
```

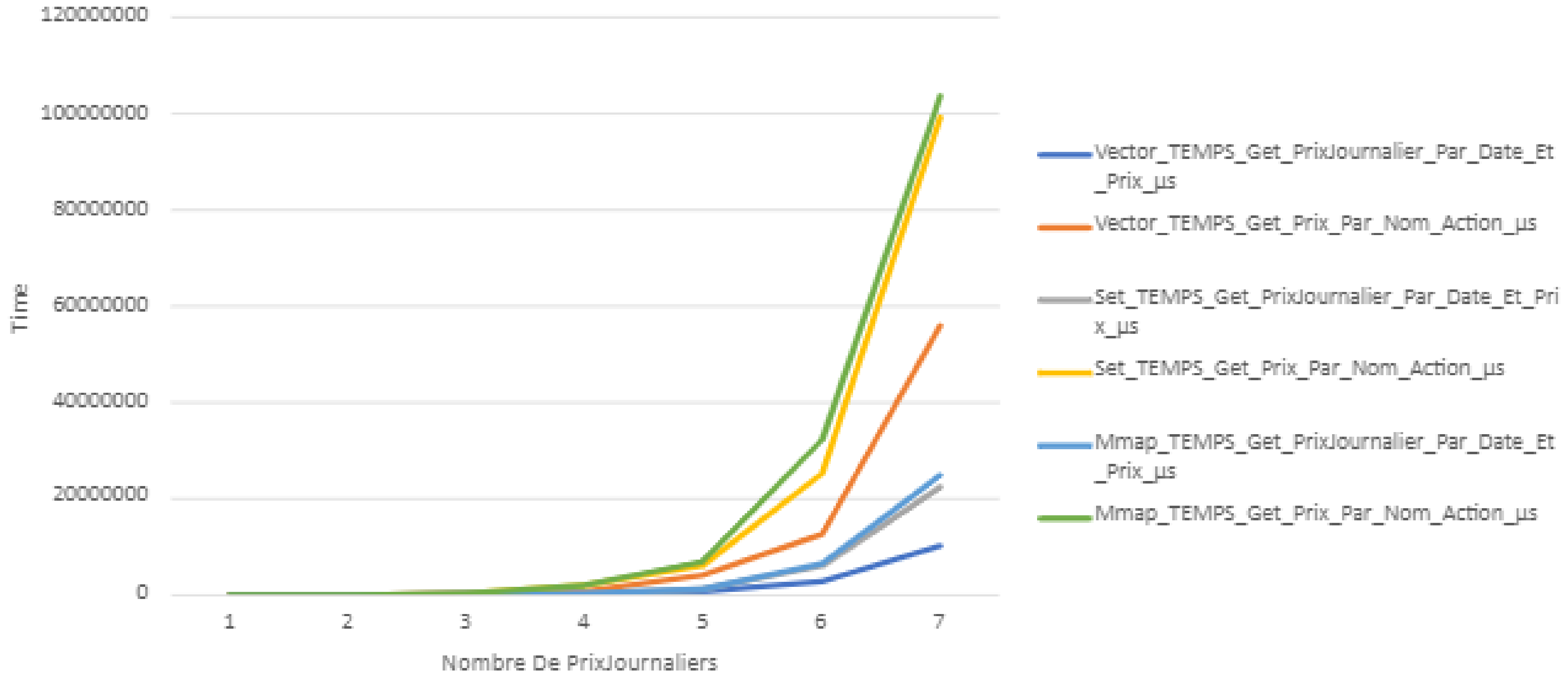
getPrixParDateEtNomAction

```
vector<double> BourseVector::getPrixParDateEtNomAction(string n) const{  
    vector<double> v={};  
    for (auto pj : this->historique) {  
        if (pj.getDate() <= (this->dateAujourd'hui) && pj.getNomAction()==n) {  
            v.push_back(pj.getPrix());  
        }  
    }  
    return v;  
}
```

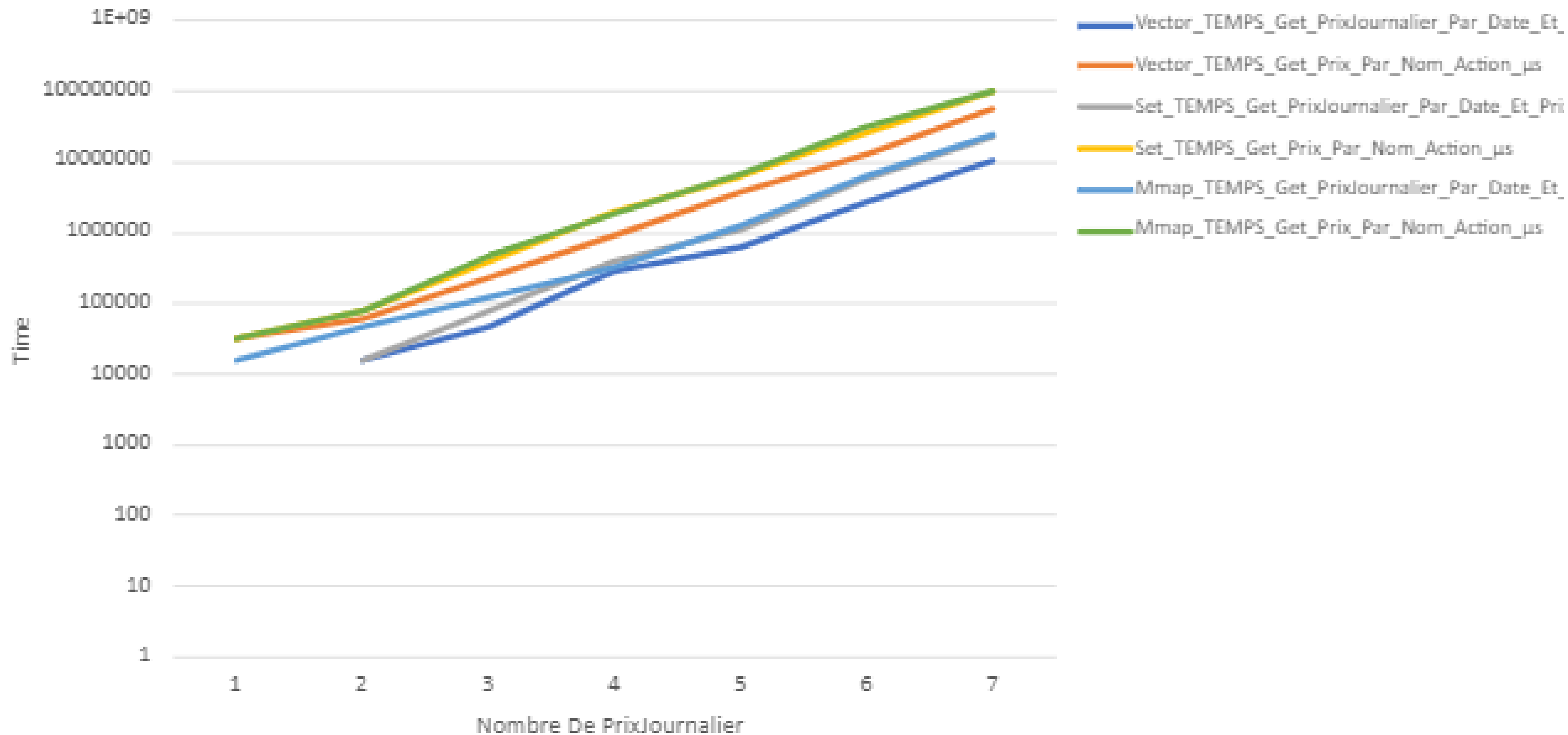
getPrixParNomAction

```
double BourseVector::getPrixParNomAction(vector<PrixJournalier> v,string n) const{
    for(auto it:v){
        if(it.getNomAction()==n){
            return(it.getPrix());
        }
    }
    return(0);
}
```


Temps De Réponses Moyens Echelle Normale



Temps De Réponses Moyens Echelle Logarithmique



STATISTIQUES DE BOURSEVECTOR :

```
Statistiques De BourseVector :  
Gain      45.2999  
MON_COMPTEUR      1  
Nbre_Transactions      65  
Nombre_d_Achats 15  
Nombre_de_Vente 50  
Rendement      0.0452999  
Solde_Final      1045.3  
TEMPS_Choisir_Transaction_µs      314210  
TEMPS_Get_PrixJournalier_Par_Date_Et_Prix_µs      51992  
TEMPS_Get_Prix_Par_Nom_Action_µs      203982  
TEMPS_SIMULATION_µs      571251  
Taux_de_Rendement_en_% 4.52999
```

STATISTIQUES DE BOURSESET :

Statistiques De BourseSet :

Gain 45.9898

MON_COMPTEUR 1

Nbre_Transactions 66

Nombre_d_Achats 11

Nombre_de_Vente 55

Rendement 0.0459898

Solde_Final 1045.99

TEMPS_Choisir_Transaction_µs 1.67831e+07

TEMPS_Get_PrixJournalier_Par_Date_Et_Prix_µs 90994

TEMPS_Get_Prix_Par_Nom_Action_µs 576973

TEMPS_SIMULATION_µs 1.75099e+07

Taux_de_Rendement_en_% 4.59898

STATISTIQUES DE BOURSEMULTIMAP :

```
Statistiques De BourseMultimap :  
Gain      6.16001  
MON_COMPTEUR      1  
Nbre_Transactions      52  
Nombre_d_Achats 11  
Nombre_de_Vente 41  
Rendement      0.00616001  
Solde_Final      1006.16  
TEMPS_Choisir_Transaction_1s      501986  
TEMPS_Get_PrixJournalier_Par_Date_Et_Prix_1s      76993  
TEMPS_Get_Prix_Par_Nom_Action_1s      339969  
TEMPS_SIMULATION_1s      923002  
Taux_de_Rendement_en_% 0.616001
```

TRADER ALGORITHMIQUE :

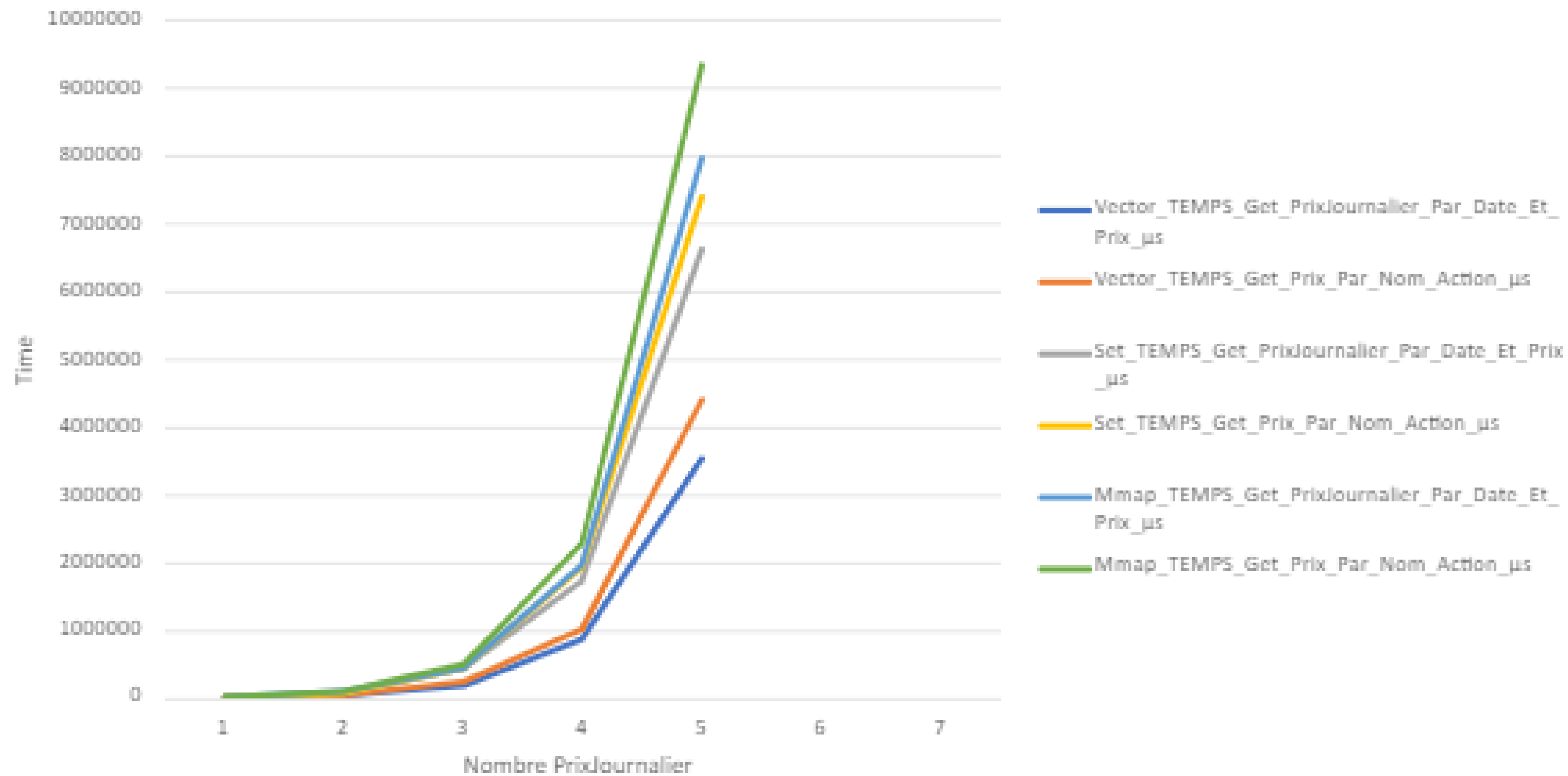
```
Transaction TraderAlgorithmique::choisirTransaction(const Bourse& bourse, const Portefeuille &portefeuille){
    Date d=bourse.getDateAujourd'hui();
    if(d.getDay()%5!=0){
        TypeTx type=Achat;
        vector<PrixJournalier> v=bourse.getPrixJournalierParDateEtPrix(portefeuille.getSolde());
        if(v.size()!=0){
            string nt=v[v.size()-1].getNomAction();
            double pr=v[v.size()-1].getPrix();
            double qte_transaction=(int)(floor((portefeuille.getSolde()/pr)));
            Titre t(nt,qte_transaction);
            Transaction T(t,type);
            return T;
        }
        else{
            Transaction T;
            return T;
        }
    }
    else{

```

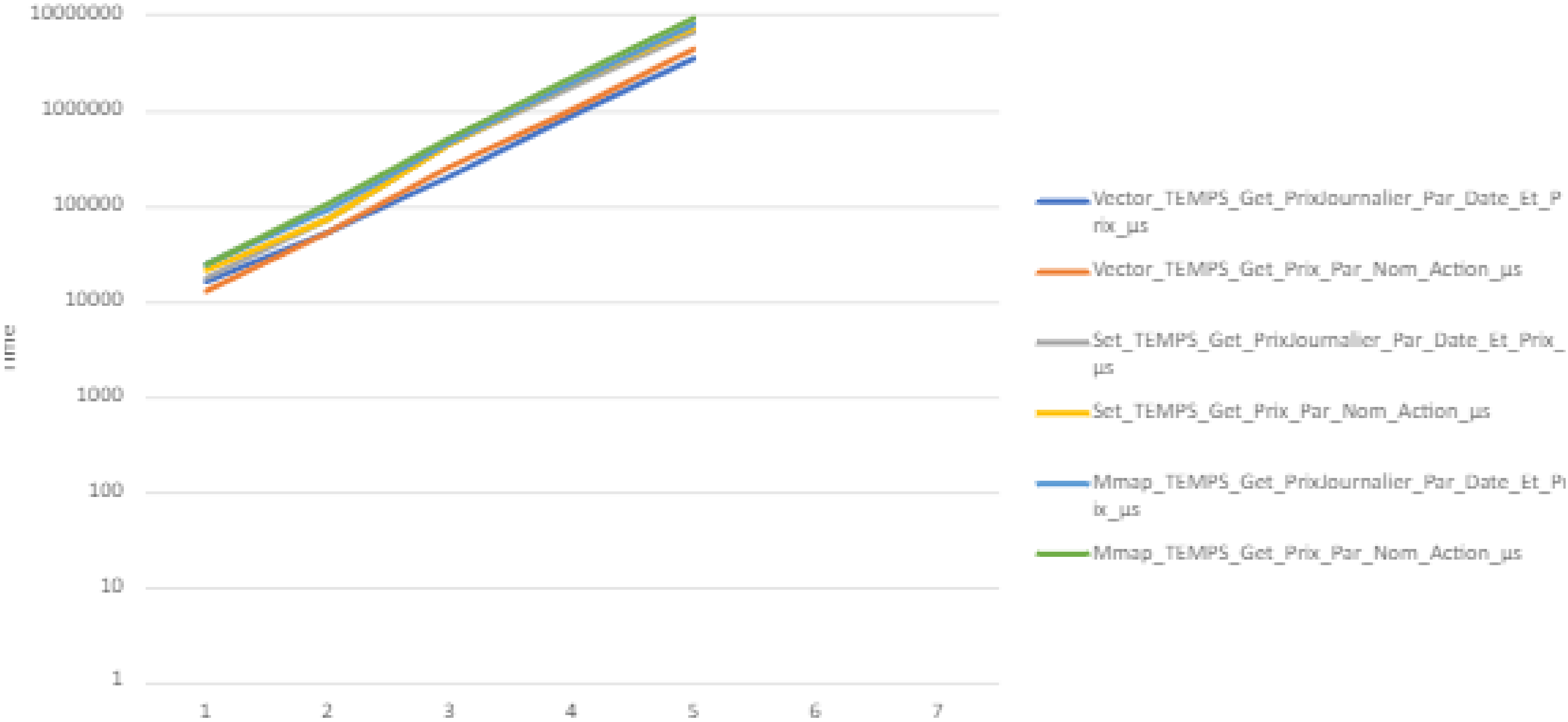
TRADER ALGORITHMIQUE :

```
else{
    if(portefeuille.getTitre().size()!=0){
        TypeTx type=Vente;
        string nt=portefeuille.getTitre()[0].getNomAction();
        double qte_transaction=(portefeuille.getTitre()[0].getQte());
        Titre t(nt,qte_transaction);
        Transaction T(t,type);
        return T;
    }
    else{
        Transaction T;
        return T;
    }
}
```


Temps De Réponses Moyens Echelle Normale D'un Traideur Algorithmique



Temps De Réponses Moyens Echelle Logarithmique D'un Traideur Algorithmique



STATISTIQUES DE BOURSEVECTOR

ALGORITHMIQUE :

```
Gain      20.6704
MON_COMPTEUR      1
Nbre_Transactions      135
Nombre_d_Achats 69
Nombre_de_Vente 66
Rendement      0.0206704
Solde_Final      1020.67
TEMPS_Choisir_Transaction_µs      652396
TEMPS_Get_PrixJournalier_Par_Date_Et_Prix_µs      228761
TEMPS_Get_Prix_Par_Nom_Action_µs      331139
TEMPS_SIMULATION_µs      1.21963e+06
Taux_de_Rendement_en_%      2.06704
```

STATISTIQUES DE BOURSESET

ALGORITHMIQUE :

```
Statistiques De BourseSet Algorithmique :  
Gain      117.34  
MON_COMPTEUR      1  
Nbre_Transactions      135  
Nombre_d_Achats 69  
Nombre_de_Vente 66  
Rendement      0.11734  
Solde_Final      1117.34  
TEMPS_Choisir_Transaction_µs      1.56093e+06  
TEMPS_Get_PrixJournalier_Par_Date_Et_Prix_µs      677974  
TEMPS_Get_Prix_Par_Nom_Action_µs      685961  
TEMPS_SIMULATION_µs      2.936e+06  
Taux_de_Rendement_en_%  11.734
```

STATISTIQUES DE BOURSE MULTIMAP

ALGORITHMIQUE :

```
Statistiques De BourseMultimap Algorithmique :  
Gain      20.6704  
MON_COMPTEUR      1  
Nbre_Transactions      135  
Nombre_d_Achats 69  
Nombre_de_Vente 66  
Rendement      0.0206704  
Solde_Final      1020.67  
TEMPS_Choisir_Transaction_µs      1.62692e+06  
TEMPS_Get_PrixJournalier_Par_Date_Et_Prix_µs      631986  
TEMPS_Get_Prix_Par_Nom_Action_µs      705959  
TEMPS_SIMULATION_µs      2.976e+06  
Taux_de_Rendement_en_%      2.06704
```

**MERCI POUR
VOTRE
ATTENTION**

