# *Diagramme des classes*

## Class Action :

string a ;

Action(string a) ;

~Action() ;

## Class Date

Int jour ;

Int mois ;

Int annee ;

Date (int jour,int mois, int annee)

~Date()

void Incrementer(int n);

void Decrementer();

void AfficherDate() ;

bool operator == (Date const& a, Date const& b0 ;

Enum TypeTransaction {achat,vente,ajouter,rien,deconnecter}

## Class Titre :

Action a ;

Int quantite ;

Titre(action a, int quantite) ;

~Titre() ;

void ModifierQuantite(int q);

Action getAction();

int getQuantite();

## Class Transaction

TypeTransaction t ;

Action a ;

Int quantite ;

Transaction (TypeTransaction a , Action b, int c) ;

~Transaction () ;

## Class prix_journalier :

Action a ;

Float prix ;

Date date ;

Prix_journalier(Action a,float prix, string date) ;

~prix_journalier()

string getAction()

Date getDate()

float getPrix()

## Class Bourse :

**Static Vect<prix_journalier> vprix;**

**Static vect<string> action;**

**Bourse() ;**

**~Bourse() ;**

**Static bool Etat(Action a ,Date d)**

**void AjouterPrixJournalier(Prix_journalier p);**

**static float TrouverPrix(string act,Date &d);**

# *Diagramme des classes*

## Class: ClientHumain

String nom;

String prénom ;

String login;

String psw;

Float budget_init ;

Float liquidité; (argent non investi)

Vect <Titre> porte;

ClientHumain (string nom, string prénom, string login, string psw,float liquidité,const float budget);

~ClientHumain ();

Ajoutertitre(Titre t, Date d) ;

Modifiertitre(Titre t, int q, Date d) ;

(q peut etre <0 en cas de vente)

AjouterBudget(float a);

static Date choisirDate();

void Acheter(Action act,int q,Date d);

void Vendre(Action act,int q,Date d);

float benefice (Date d);

getBudget();

getLiquidite ();

getLogin () ;

get Psw() ;

# *Diagramme des classes*

## Class Trader

String nom;

String prénom ;

String login;

String psw;

Float budget_init ;

Float liquidite ;

vector<Titre>porte;

virtual vector<Transaction> Trans(Date cejour);

Trader (string nom, string prénom, string login, string psw,float liquidité,const float budget);

Virtual ~Trader()

Static Date choisirDate() ;

bool Test (string nom);

void AjouterTitre(Titre t, Date d);

void ModifierTitre(Action act,int q, Date d);

void AjouterBudget(float a);

float Benefice(Date cejour);

void Appliquer (Date cejour,bool &connecte, vector<transaction> tr);

string getLogin () ;

string getPsw() ;

float getLiquidite () ;

virtual vector <Transaction> Trans(Date cejour)=0 ;

vector <Titre>porte ;

## Class RobotTrader

**Robotrader (string a, string b, string c, string d, float e);**

**Virtual** ~Robotrader () ;

Virtual vector <Transaction> Trans(Date &cejour) ;

Vector <Transaction> Vendrerobo (Date &cejour) ;

Vector <Transaction> Achatrobot (Date &cejour) ;

## Class RobotTrader2

**Robotrader2 (string a, string b, string c, string d, float e);**

**Virtual** ~Robotrader2 () ;

Virtual vector <Transaction> Trans(Date &cejour) ;

Vector <Transaction> Vendrerobo (Date &cejour) ;

Vector <Transaction> Achatrobot (Date &cejour) ;