C프로그래밍및실습

암호화폐거래 pnl

최종 보고서

제출일자:23.12.10

제출자명:공가웅

제출자학번:233911

1.프로젝트 목표

1) 배경 및 필요성

암호화폐 거래를 할 때 현재 포지션의 수익률은 명시되어 알 수 있지만 그 날의 총 수익률 및 그 달의 총 수익률은 명시되어 있지 않아 어느정도의 이득과 손실을 보았는지 알기 어려움. 이를 해결하기위해 pnl(profit and loss)수익률을 고객에게 명시되어지도록 하는 프로그램이 필요함.

2) 프로젝트 목표

고객들의 투자 수익률을 명시하여 현재 투자 방식이 괜찮은지 판단할 수 있도록 함.

3) 차별점

몇몇의 거래소들은 pnl수익률이 명시되어 있지 않아 현재까지의 종합적인 수익률 및 손실율을 입금된 금액과 현재 금액의 차익을 통해 계산을 해야했던 반면 이 프로그램을 이용하여 이 과정을 없애고 바로 알 수 있도록함.

2. 기능 계획

1) 기능 1

- 거래의 성공률을 보는 기능

2) 기능 2

- 고객의 당일 pnl값을 보는 기능

3) 기능 3

- 고객의 total pnl값을 보는 기능

4) 기능 4

- 고객의 포지션 종료시 해당 포지션의 수익률을 보는 기능

5) 기능 5

- 고객의 포지션 시작시 coin종류, long, short 배팅과 설정 배율을 지정하는 기능

6) 기능 6

- coin종류에 따라 주문수를 나누어 coin종류당 비율을 명시하는 기능

3. 기능 구현

(1) initializeTradeInfo

- 저장한 구조체의 배열의 동적 할당을 하기 위한 기능

- 문자열은 동적으로 저장할 때 strcpy함수를 통해 저장

- 참조연산자를 통해 함수를 더 간단하게 함

-코드 스크린샷

```
void initializeTradeInfo(struct TradeInfo* trade) {
    printf("거래시 coin종류를 입력하세요(BTC,ETH,XRP(대문자로 입력)): ");
    char temp[100];
    scanf_s("%s", temp, (int)sizeof(temp));
    trade->CoinType = (char*)malloc((strlen(temp) + 1) * sizeof(char));
    strcpy_s(trade->CoinType, strlen(temp) + 1, temp);

    printf("거래시 Long/Short정보를 입력하세요(대문자로 입력): ");
    scanf_s("%s", temp, (int)sizeof(temp));
    trade->CoinBuySell = (char*)malloc((strlen(temp) + 1) * sizeof(char));
    strcpy_s(trade->CoinBuySell, strlen(temp) + 1, temp);

    printf("거래시 구매량(KRW)을 입력하세요: ");
    scanf_s("%d", &trade->Coinvalue);

    printf("해당거래에 사용한 배율을 입력하세요: ");
    scanf_s("%d", &trade->Magnification);

    printf("해당거래의 시작가와 종료가를 입력하세요: ");
    scanf_s("%lf %lf", &trade->StartPrice, &trade->EndPrice);
}
```

(2) CoinProfit

- 거래당 수익률을 보기위한 기능

- a,b의 크기에 따른 조건을 주어 해당 식의 값을 profit1,2에 리턴되도록 함

- 코드 스크린샷

```
double CoinProfit(const struct TradeInfo* trade, int i) {
    double profit1, profit2;
    double a, b;
    int c;
    a = trade[i - 1].StartPrice;
    b = trade[i - 1].EndPrice;
    c = trade[i - 1].Magnification;
    if (a <= b) {
        profit1 = (double)c * (((b - a) / a) * 100);
        return profit1;
    }
    else if (a > b) {
        profit2 = (double)c * (((a - b) / a) * 100);
        return profit2;
    }
}
```

(3) CoinTypeRate

- 거래된 coin종류의 비율을 보는 기능

- 각 조건을 구조체에 저장된 문자에 따라 다르게 두고 각각 횟수가 카운트되어 비율을 출력하도록함

- 코드 스크린샷

```c
void CoinTypeRate(struct TradeInfo* trade, int numTrades) {
    int a = 0;
    int b = 0;
    int c = 0;
    double d, e, f;
    for (int i = 0; i < numTrades; i++) {
        if (*trade[i].CoinType == 'B') {
            a += 1;
        }
        else if (*trade[i].CoinType == 'E') {
            b += 1;
        }
        else if (*trade[i].CoinType == 'X') {
            c += 1;
        }
    }
    d = (double)a / (a + b + c) * 100;
    e = (double)b / (a + b + c) * 100;
    f = (double)100 - (d + e);
    printf("BTC:%.1lf%c, ETH:%.1lf%c, XRP:%.1lf%c₩n", d, '%', e, '%', f, '%');
}
```

(4) WinRate

- 고객이 한 거래의 승률을 보는 기능

- 각 거래에 이득이 발생했을 시 1씩더해주는 조건을 통해 a가 1씩 더해지도록 하여 b에 저장된 식을 통해 승률을 보여줌

-코드 스크린샷

```c
void WinRate(struct TradeInfo* trade, int num) {
    int a = 0;
    for (int i = 0; i < num; i++) {
        if (trade[i].StartPrice < trade[i].EndPrice) {
            if (*trade[i].CoinBuySell == 'L') {
                a++;
            }
        }
        else if (trade[i].StartPrice >= trade[i].EndPrice) {
            if (*trade[i].CoinBuySell == 'S') {
                a++;
            }
        }
    }
    double b = (double)a / num * 100;
    printf("고객님의 총거래 승률은 %.1lf%c입니다", b, '%');
}
```

(5) RecentPnl

- 고객의 당일 거래의 수익과 수익률을 명시하는 기능

- 당일 몇번의 거래가 있었는지 입력받은 후 그 만큼 반복하는 기능을 만듦, 각 조건에 따라 a가 더해지거나 빼자는 기능을 만들고 마지막에 b 식을 통해 당일의 수익금과 수익률을 명시함

-코드 스크린샷

```
void RecentPnl(struct TradeInfo* trade, int num, double AV) {
    double a = 0;//수익금
    double b;//수익률
    for (int i = 0; i < num; i++) {
        if (trade[i].StartPrice < trade[i].EndPrice) {
            if (*trade[i].CoinBuySell == 'L') {
                a += (double)trade[i].Coinvalue * (((CoinProfit(trade, i + 1)) / 100));
            }
            else if (*trade[i].CoinBuySell == 'S') {
                a -= (double)trade[i].Coinvalue * (((CoinProfit(trade, i + 1)) / 100));
            }
        }
        else if (trade[i].StartPrice >= trade[i].EndPrice) {
            if (*trade[i].CoinBuySell == 'S') {
                a += (double)trade[i].Coinvalue * (((CoinProfit(trade, i + 1)) / 100));
            }
            else if (*trade[i].CoinBuySell == 'L') {
                a -= (double)trade[i].Coinvalue * (((CoinProfit(trade, i + 1)) / 100));
            }
        }
    }
    b = (double)a / AV * 100;
    printf("고객님의 금일Pnl:%.1lfKRW(%.1lf%c)", a + AV, b, '%');
}
```

(6) TotalPnl

- 고객의 총 거래의 수익과 수익률을 명시하는 기능

- 동적 배열에 할당이 이루어진 만큼 반복하는 기능을 만듦, 각 조건에 따라 a가 더해지거나 빼자는 기능을 만들고 마지막에 b 식을 통해 당일의 수익금과 수익률을 명시함

- 코드 스크린샷

```c
void TotalPnl(struct TradeInfo* trade, int num, double AV) {
    double a = 0;//수익금
    double b;//수익률
    for (int i = 0; i < num; i++) {
        if (trade[i].StartPrice < trade[i].EndPrice) {
            if (*trade[i].CoinBuySell == 'L') {
                a += (double)trade[i].Coinvalue * (((CoinProfit(trade, i + 1)) / 100));
            }
            else if (*trade[i].CoinBuySell == 'S') {
                a -= (double)trade[i].Coinvalue * (((CoinProfit(trade, i + 1)) / 100));
            }
        }
        else if (trade[i].StartPrice >= trade[i].EndPrice) {
            if (*trade[i].CoinBuySell == 'S') {
                a += (double)trade[i].Coinvalue * (((CoinProfit(trade, i + 1)) / 100));
            }
            else if (*trade[i].CoinBuySell == 'L') {
                a -= (double)trade[i].Coinvalue * (((CoinProfit(trade, i + 1)) / 100));
            }
        }
    }
    b = (double)a / AV * 100;
    printf("고객님의 총Pnl:%.1lfKRW(%.1lf%c)", a + AV, b, '%');
}
```

4. 테스트 결과

(1) initializeTradeInfo

- 저장한 구조체의 배열의 동적 할당을 하기 위한 기능

- 테스트 결과 스크린샷

```
계좌의 초기금액을 입력하세요(KRW).
10000
몇 번의 거래를 입력하시겠습니까?
2

1번째 거래의 정보를 입력하세요.
거래시 coin종류를 입력하세요(BTC,ETH,XRP(대문자로 입력)): BTC
거래시 Long/Short정보를 입력하세요(대문자로 입력): LONG
거래시 구매량(KRW)을 입력하세요: 1000
해당거래에 사용한 배율을 입력하세요(1~100): 10
해당거래의 시작가와 종료가를 입력하세요: 40000 41000

2번째 거래의 정보를 입력하세요.
거래시 coin종류를 입력하세요(BTC,ETH,XRP(대문자로 입력)): XRP
거래시 Long/Short정보를 입력하세요(대문자로 입력): SHORT
거래시 구매량(KRW)을 입력하세요: 1000
해당거래에 사용한 배율을 입력하세요(1~100): 15
해당거래의 시작가와 종료가를 입력하세요: 5000 4990
```

(2) CoinProfit

- 거래당 수익률을 보기위한 기능

-코드 스크린샷

```
1.해당 거래의 수익률
2.coin종류당 거래횟수
3.내 거래 성공률
4.당일 pnl보기
5.total pnl 보기0.프로그램 종료


이용할 옵션을 선택하세요.(0~5): 1
1번째 옵션을 선택하셨습니다.
몇 번째 거래의 수익률을 볼건지 입력하시오: 1
해당 거래의 수익률은 +25.0%입니다.


이용할 옵션을 선택하세요.(0~5): 1
1번째 옵션을 선택하셨습니다.
몇 번째 거래의 수익률을 볼건지 입력하시오: 2
해당 거래의 수익률은 +3.0%입니다.
```

(3) CoinTypeRate

- 거래된 coin종류의 비율을 보는 기능

-코드 스크린샷

```
이용할 옵션을 선택하세요.(0~5): 2
2번째 옵션을 선택하셨습니다.

coin종류당 거래비율은 다음과 같습니다.
BTC:50.0%, ETH:0.0%, XRP:50.0%
```

(4) WinRate

- 고객이 한 거래의 승률을 보는 기능

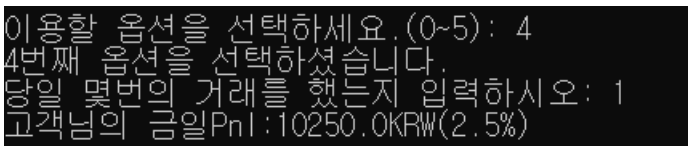
-코드 스크린샷

```
이용할 옵션을 선택하세요.(0~5): 3
3번째 옵션을 선택하셨습니다.
고객님의 총거래 승률은 100.0%입니다
```

(5) RecentPnl

- 고객의 당일 거래의 수익과 수익률을 명시하는 기능

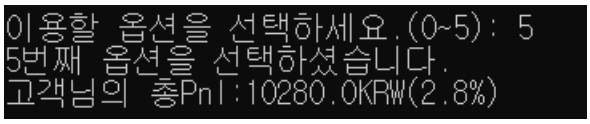
-코드 스크린샷



(6) TotalPnl

- 고객의 총 거래의 수익과 수익률을 명시하는 기능

-코드 스크린샷



5. 계획 대비 변경 사항

1) CoinTypeRate

- 이전: coin종류당 거래횟수 명시

- 이후: coin종류당 거래비율 명시

- 사유: 대부분의 거래소에서 거래비율을 명시해주는 기능을 씀

6. 느낀점

- 실기시험을 보는 것 보다 프로젝트를 직접 진행해보는게 실력향상에 훨씬 도움이 되었음