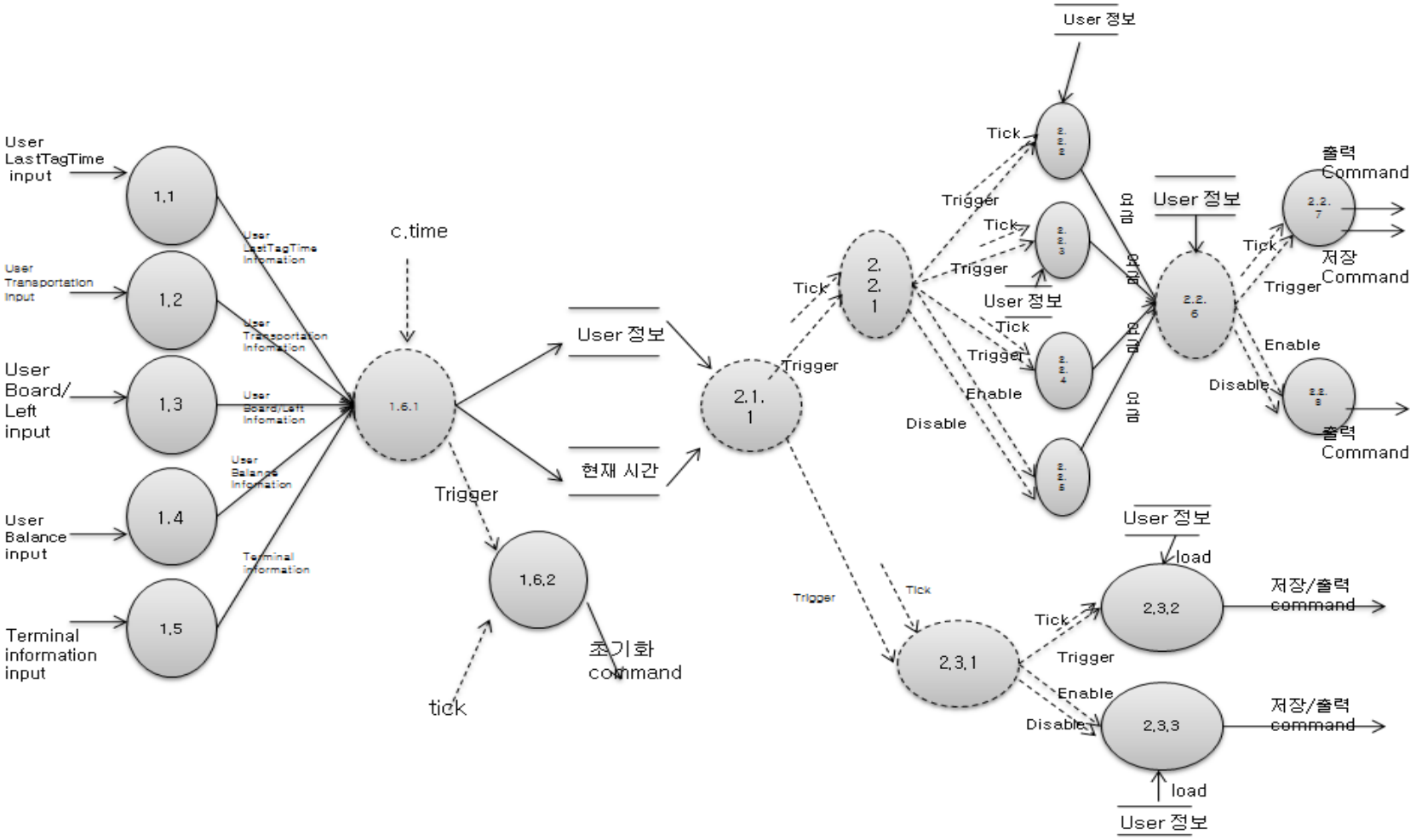
교통카드 시스템

Team Presentation #1

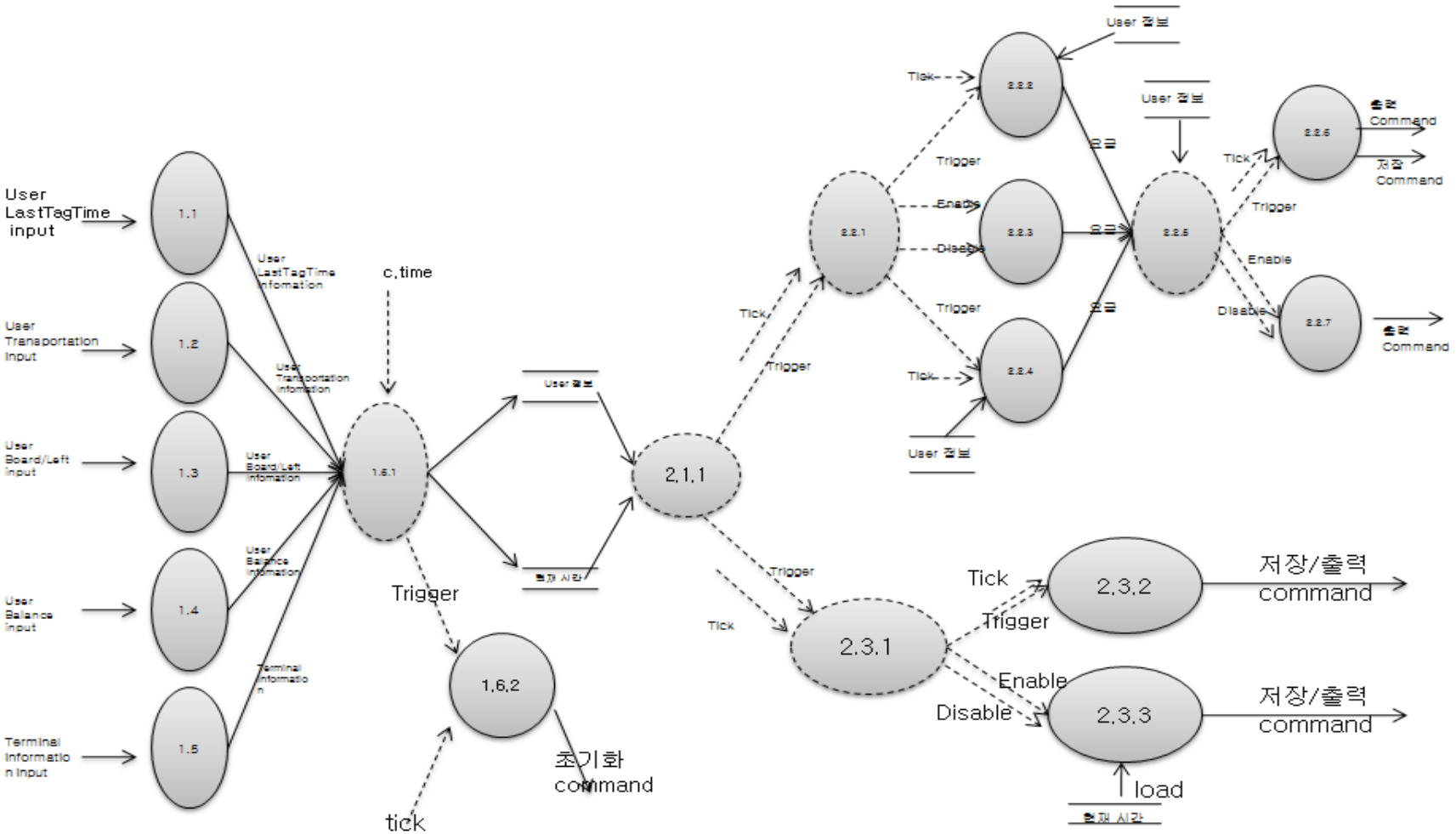
TEAM[T4]

노은방 200811428
김상민 200910044
박수민 201111353
한별 201214217

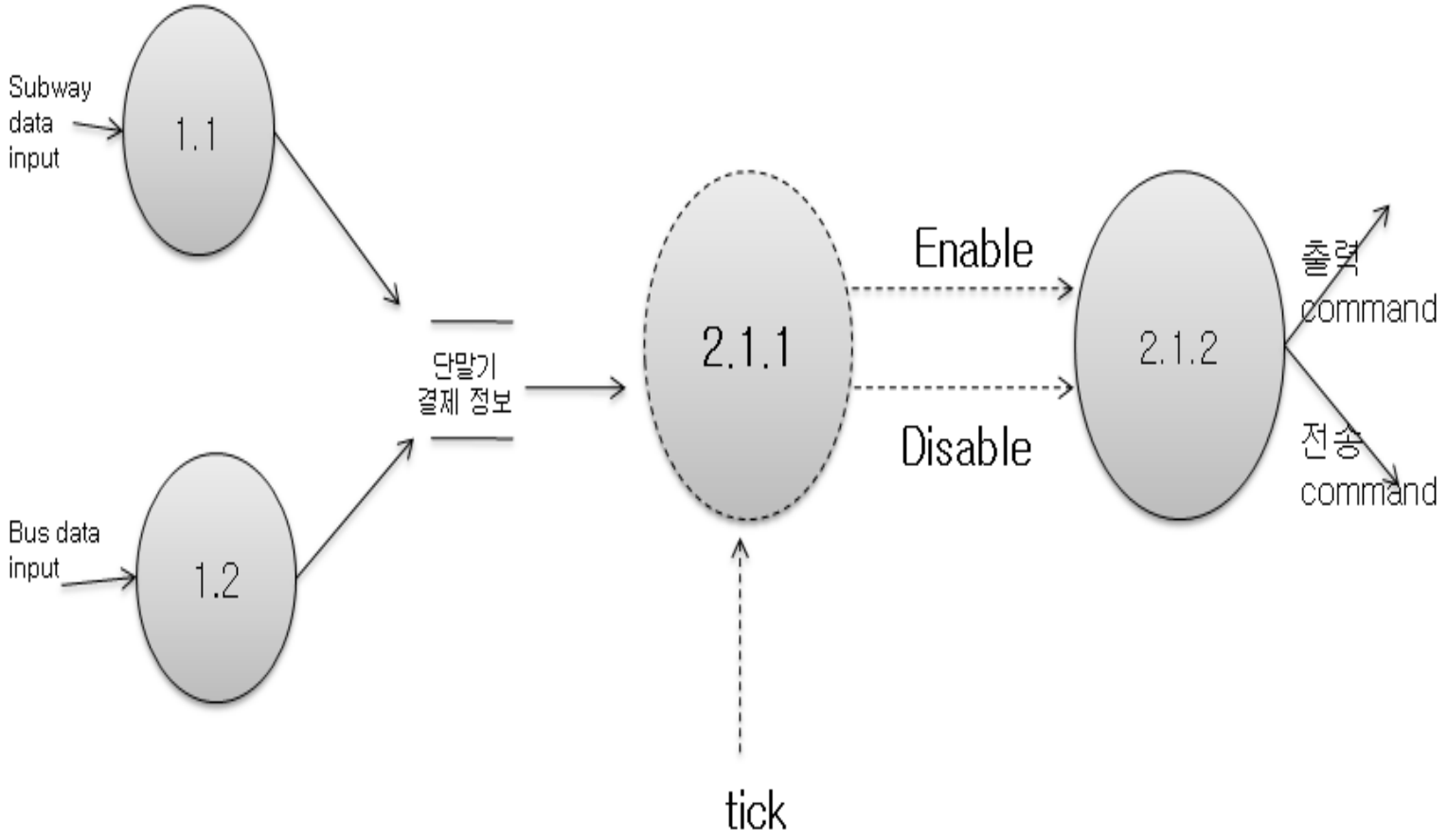
Structured Analysis (지하철)
DFD - Overall



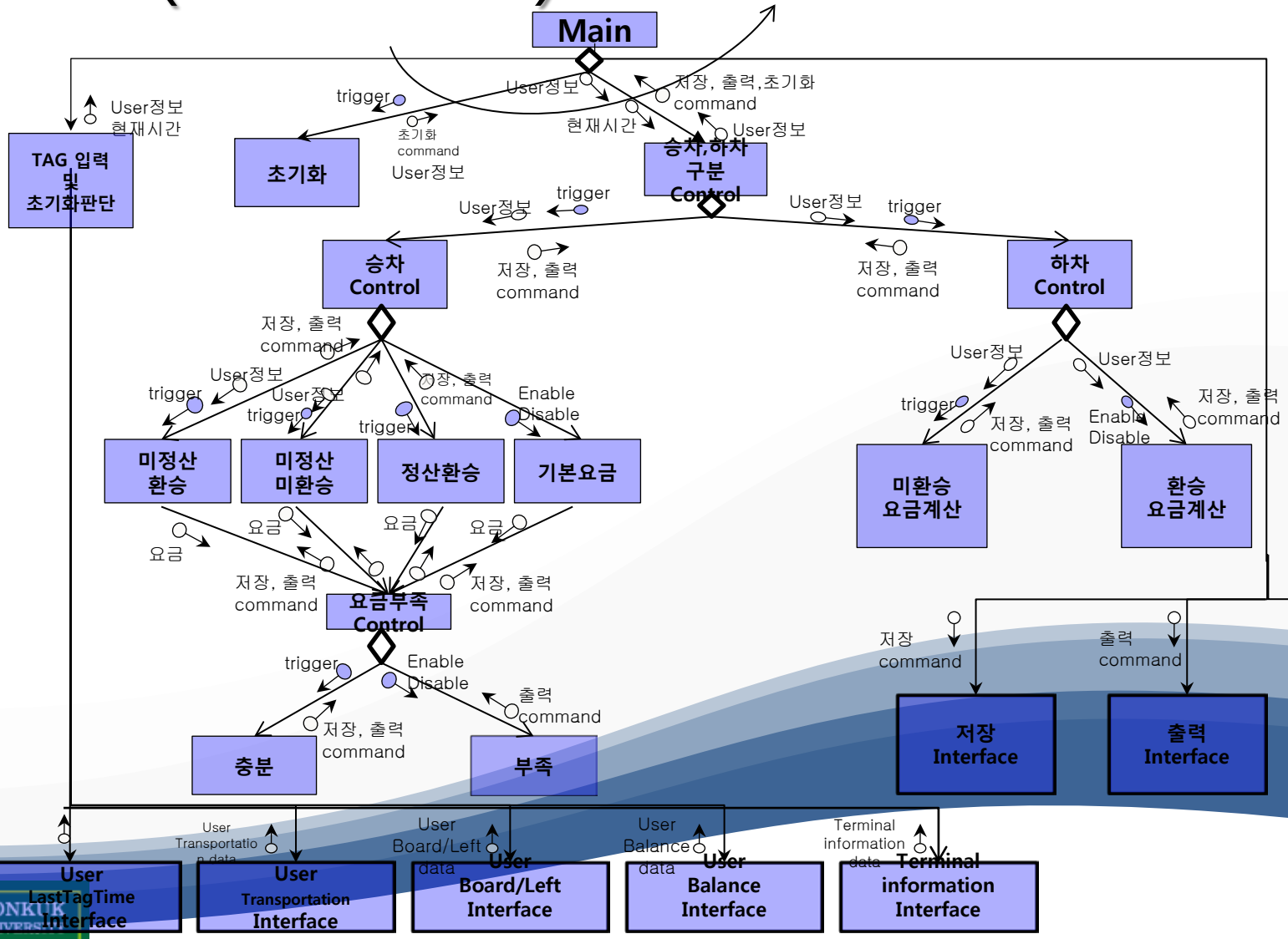
Structured Analysis (버스)
DFD - Overall



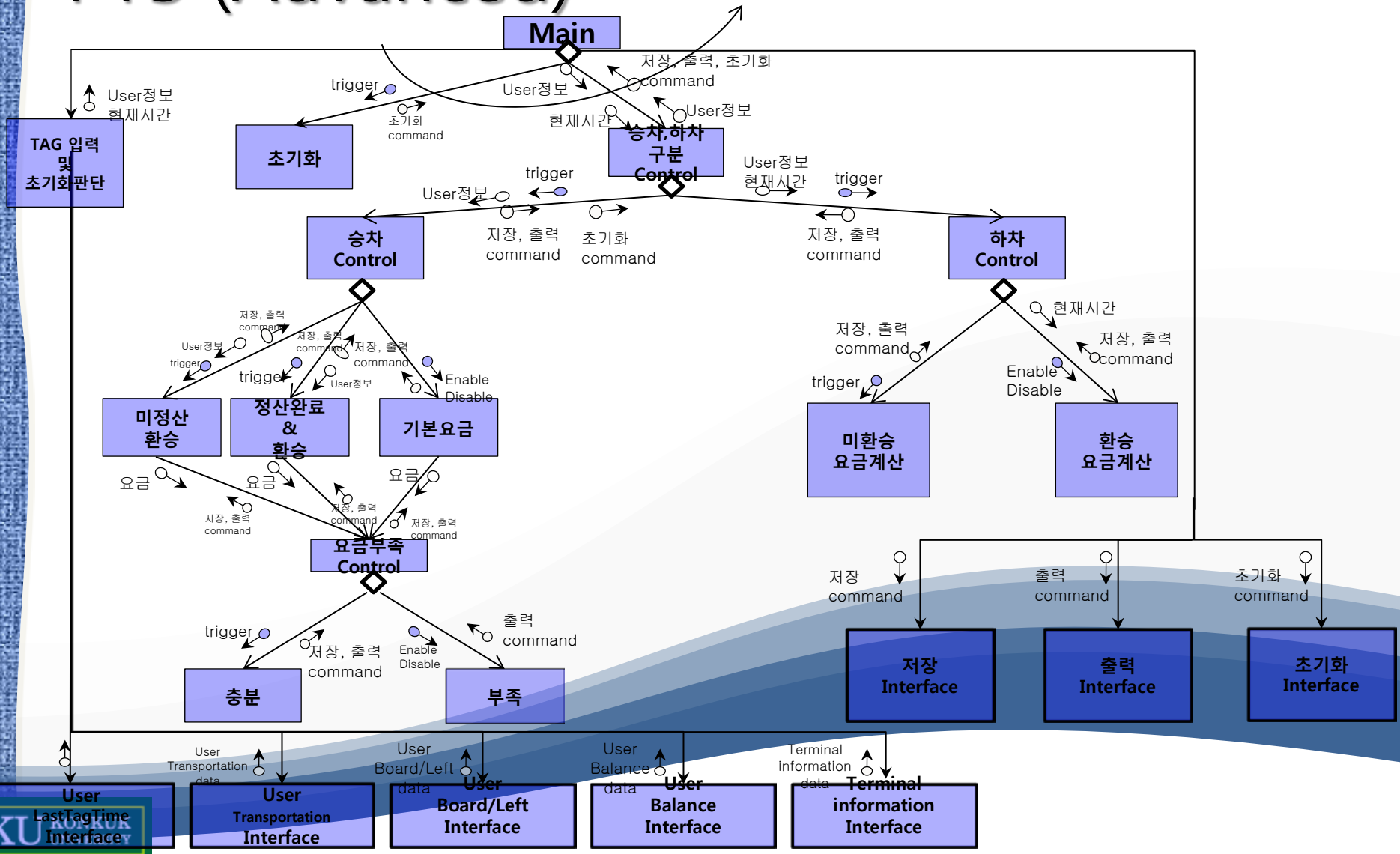
Structured Analysis (정산시스템)
DFD - Overall



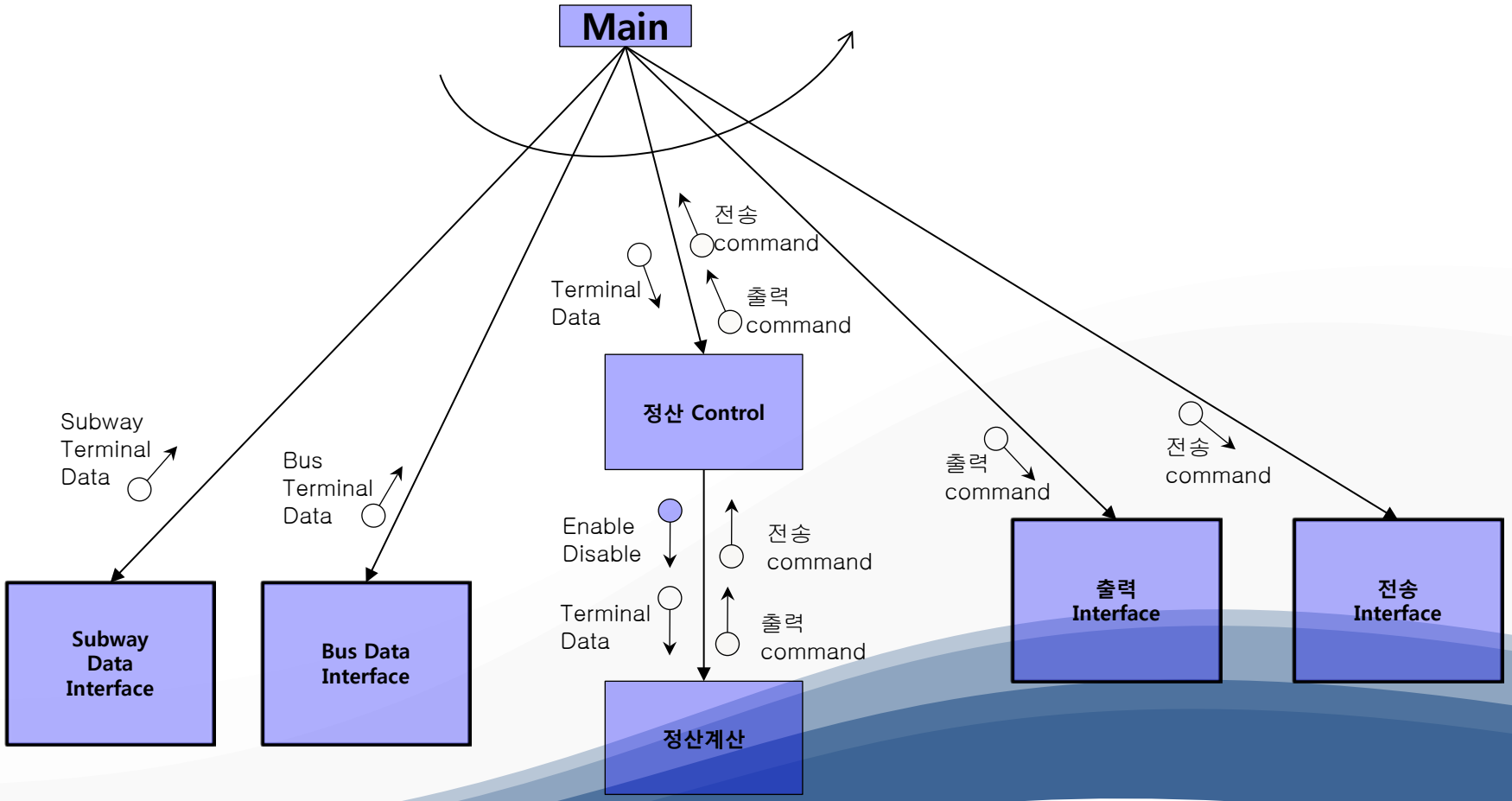
Structured Chart (지하철)
PTS (Advanced)



Structured Chart (버스)
PTS (Advanced)



Structured Chart (정산)
PTS (Advanced)



Development Environment

- Windows 7, Windows 8.1
- Cygwin
- Visual studio 2010, 2013

Project Structure (지하철)

- Header
  - #include "terminal.h"
  - #include "subway.h"
  - #include "subway_in.h"
  - #include "subway_out.h“
  - #include <stdio.h>
  - #include <string.h>
  - #include <stdlib.h>
  - #include <time.h>
  - #include <Windows.h>
  - #include <conio.h>

Project Structure (지하철)

- Code
  - main.c
  - terminal.c
  - subway.c
  - subway_in.c
  - subway_out.c

Project Structure (버스)

- Header
  - #include "busInput.h"
  - #include "busbus.h"
  - #include <stdio.h>
  - #include <string.h>
  - #include <stdlib.h>
  - #include <conio.h>
  - #include <time.h>
  - #include <Windows.h>
- Code
  - busInput.c
  - busbus.c

Project Structure (정산)

- Header
  - #include <stdio.h>
  - #include <string.h>
  - #include <stdlib.h>
  - #include <math.h>
  - #include "getinfo.h"
- Code
  - main.c

Definition (지하철)

⊙ #define BASIC_FEE 1050

```
#define BASIC_FEE 1050
#define MAX_USER_INPUT 50
#define BUS 0
#define METRO 1
#define IN 0
#define OUT 1

int money;
char c;
int terminalCount[6];
int check;
```

```
int in_out;
int balance;
char terminalinfo[6];
}userinfo;

struct currentInfo{
char lastTagTime[20];
int tranaportation;
int in_out;
int balance;
char terminalinfo[6];
}currentinfo;
```

⊙ struct userInfo

⊙ struct currentInfo

Definition (버스)

- #c
- #c
- #c
- #c
- st
- st
- int

```
#define BUS 0
#define METRO 1
#define IN 0
#define OUT 1

struct userInfo{
char lastTagTime[20];
int transportation;
int in_out;
int balance;
char terminalinfo[6];
};
struct currentInfo{
char lastTagTime[20];
int transportation;
int in_out;
int balance;
char terminalinfo[6];
};
int terminalCount=0;
```

Definition (정산)

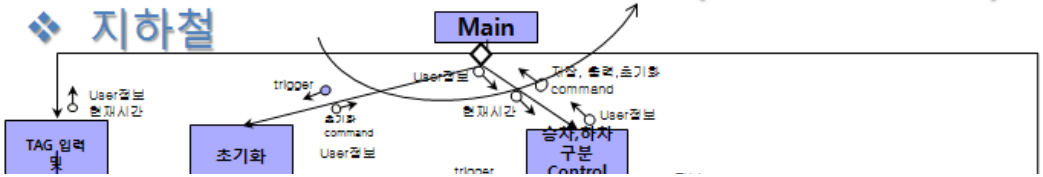
```
#define MAX_SIZE 100
#define MAX_INPUT_SIZE 50
#define BUS 0
#define METRO 1
#define IN 0
#define OUT 1

int leng;
int rleng;          //result 배열의 총 크기
struct termInfo{
        char lastTagTime[20];
        int transportation;
        int in_out;
        int balance;
        char terminalinfo[6];
};

struct result{
        char lastTagTime[20];
        int transportation;
        int fee;
        char terminalinfo[6];
};
```

Code Analysis(지하철)

Code Analys

Structured Chart – PTS (Advanced)

❖ 지하철




```
terminal.c
//유저정보는한줄로입력받아서처리
char* getLastTagTime(){
FILE *f;/*유저태그파일한줄에한번의태그정보모두입력*/
char userInput[MAX_USER_INPUT];
char tagTime[20];
f = fopen("C:\\수민\\test\\user.txt", "r");
if (f == NULL) perror("사용자가없습니다");
else{
while (fgets(userInput, MAX_USER_INPUT, f) != NULL){}
fclose(f);
}
/*userInput 형태유저카드정보에type들한줄로이어져있음.*/
/*tagTime[12] = '\0';
strncpy(tagTime,userInput,12);*/
strcpy(tagTime, strtok(userInput, " "));
printf("lasttagtime : %s\n", tagTime);
return tagTime;
}
// 태그된시간찾아서반환해주는함수
```

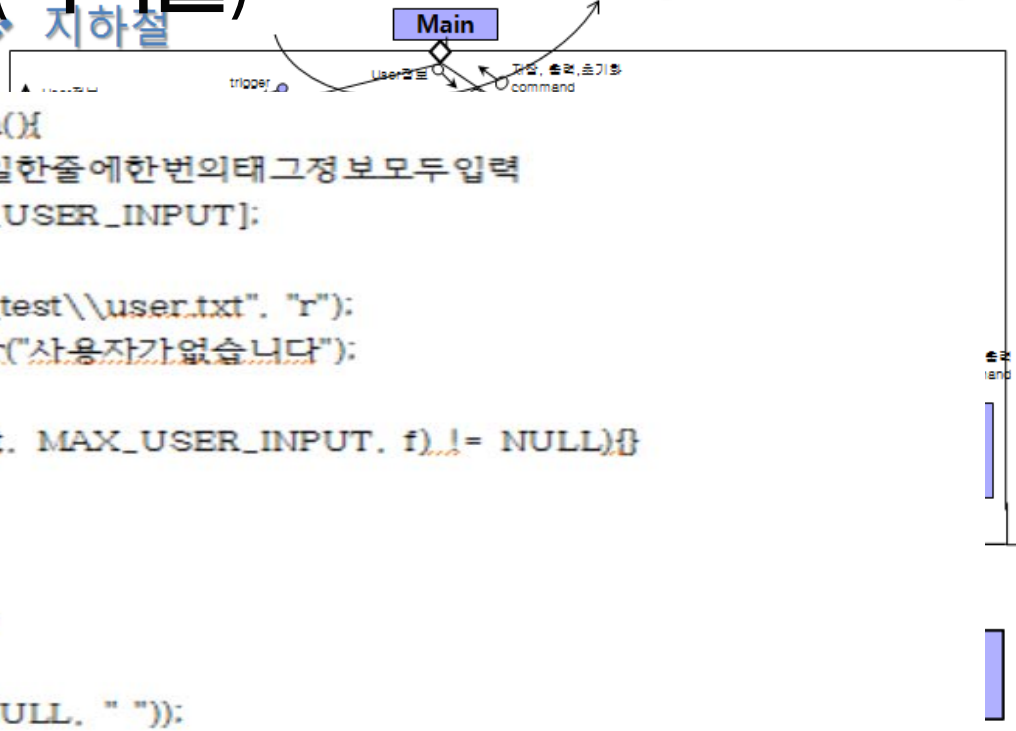
초기화 command → 초기화 Interface

Reference No.	
Name	
Input	
Output	
Process Description	입력 사항에 카드에서 마지막으로 태그한 시간 정보를 전달한다.

Code Analysis(지하철)

Structured Chart – PTS (Advanced)

❖ 지하철



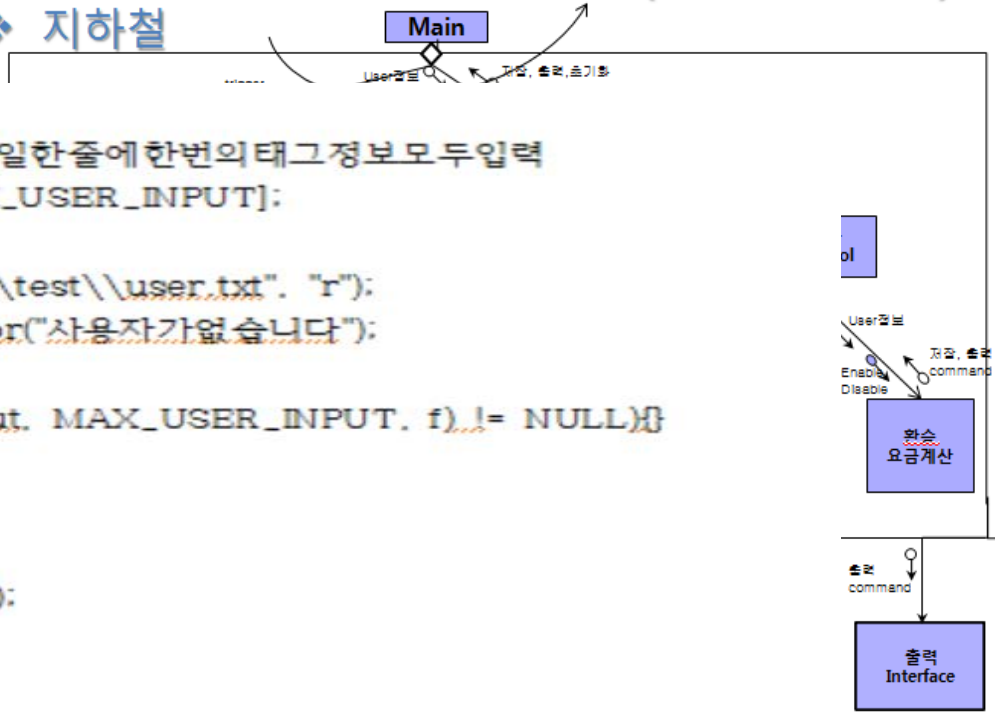

```
int getTransportation(){
FILE *f://유저태그파일한줄에한번의태그정보모두입력
char userInput[MAX_USER_INPUT];
char temp[6];
f = fopen("C:\\수민\\test\\user.txt", "r");
if (f == NULL) perror("사용자가없습니다");
else{
while (fgets(userInput, MAX_USER_INPUT, f) != NULL){}
fclose(f);
}
//파일처리구간
strtok(userInput, " ");
//태그된시간
strcpy(temp, strtok(NULL, " "));
//교통수단
printf("transport : %s\n", temp);
if (strcmp(temp, "BUS") == 0)
return BUS;
else if (strcmp(temp, "METRO") == 0)
return METRO;
else
return METRO;
}
// 버스면0반환, 지하철이면1반환, 최초탑승이면이전탑승수단은지하철로정한다.
// 지하철터미널에선지하철이고하차인경우기본금액
```

Reference No.
Name
Input
Output
Process Descripti

Code Analys

Structured Chart – PTS (Advanced)

❖ 지하철



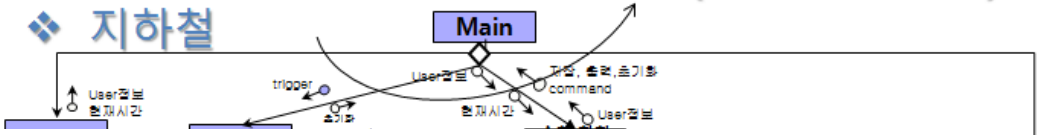

```
int getin_out() {
FILE *f://유저태그파일한줄에한번의태그정보모두입력
char userInput[MAX_USER_INPUT];
char temp[4];
f = fopen("C:\\수민\\test\\user.txt", "r");
if (f == NULL) perror("사용자가없습니다");
else{
while (fgets(userInput, MAX_USER_INPUT, f) != NULL){}
fclose(f);
}
//파일처리구간
strtok(userInput, " ");
//태그된시간
strtok(NULL, " ");
//교통수단지나고
strcpy(temp, strtok(NULL, " "));
printf("inout:%s\n", temp);
if (strcmp(temp, "IN") == 0)
return IN;
else if (strcmp(temp, "OUT") == 0)
return OUT;
else
return OUT;
}
// 유저카드에탑승정보가승차면0 하차면1 최초면하차이다.
// 버스터미널에선이전에버스에서하차한경우에기본금액이기에.
```

Reference No.
Name
Input
Output
Process Description

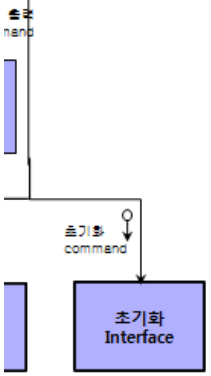
Code Analys

Structured Chart – PTS (Advanced)

❖ 지하철




```
int getBalance() {
FILE *f://유저태그파일한줄에한번의태그정보모두입력
char userInput[MAX_USER_INPUT];
char* balance;
int i = 0;
f = fopen("C:\\수민\\test\\user.txt", "r");
if (f == NULL) perror("사용자가없습니다");
else{
while (fgets(userInput, MAX_USER_INPUT, f) != NULL){}
fclose(f);
}
//파일처리구간
strtok(userInput, " ");
//태그된시간
for (i = 0; i<2; i++)
strtok(NULL, " ");
//교통수단, 승차하차
balance = strtok(NULL, " ");
printf("balance: %s\n", balance);
return atoi(balance);
}
//잔액반환
```

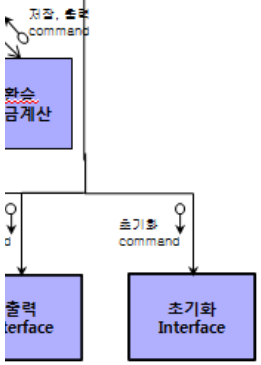


Reference No.	
Name	
Input	
Output	
Process Descripti...	

Code Analysis – Structured Chart – PTS (Advanced)

❖ 지하철


```
char* getTerminalinfo(){
FILE *f://유저태그파일한줄에한번의태그정보모두입력
char userInput[MAX_USER_INPUT];
char terminalinfo[6];
int i = 0;
f = fopen("C:\\수민\\test\\user.txt", "r");
if (f == NULL) perror("사용자가없습니다");
else{
while (fgets(userInput, MAX_USER_INPUT, f) != NULL){}
fclose(f);
}
//파일처리구간
strtok(userInput, " ");
//태그된시간
for (i = 0; i<3; i++)
strtok(NULL, " ");
//교통수단, 승차하차, 잔액
strcpy(terminalinfo, strtok(NULL, " "));
printf("terminal:%s\n", terminalinfo);
return terminalinfo;
}
//탑승단말기정보반환
```



Reference No.	
Name	
Input	
Output	
Process Description	

Code Analysis

Structured Chart – PTS (Advanced)

❖ 지하철

Main

```
void tagInput(){
time_t timer;//현재시간
time_t initTimer;//시스템최초시작시간
struct tm *t;
char tempSecond[3];// 초를char 배열로
char tempMinute[3];// 분을char 배열로
char cterminalCount[3];//단말기정보의count 값을char로저장해줌
char useMetro = 'z';//지하철이용하는지확인
char metrob[3] = "0_";//지하철사용
int secTime = 0;//시스템시작으로부터현재시간초단위차이
int min;
int sec;
int secWait = 0;
int kb = 0;
int tempi = 0;//in out 버퍼
int initializer = 1;//초기화여부확인
int ios;//승차하차입력버퍼
initTimer = time(NULL);//시간을초단위로받음
t = localtime(&initTimer);//시간의시분초분할
while (1){
printf("태그할경우t 입력:");
printf("\n");
while (1){
tempi = 0;
timer = time(NULL);//시간을초단위로받음
t = localtime(&timer);//시간의시분초분할
secTime = timer - initTimer;//시스템시작부터지금까지시간
if (kb == 1)
{
secWait = secTime - secWait;
kb = 0;
}
```

```
secTime -= secWait;
min = secTime / 60;
sec = secTime % 60;
sprintf(tempSecond, "%d", sec);//태그시간에분입력
sprintf(tempMinute, "%d", min);//태그시간에초입력
strcpy(currentinfo.lastTagTime, tempMinute);
strcat(currentinfo.lastTagTime, ":");
strcat(currentinfo.lastTagTime, tempSecond);
printf("%02d:%02d\n", min, sec);

if (kbhit()){
secWait = secTime;
scanf("%c", &useMetro);
fflush(stdin);
kb = 1;
if (useMetro == 't'){
system("cls");
/*printf("\nTAG TIME %02d:%02d\n", min, sec);*/
break
}
}
Sleep(1000);
/*break;*/
}
```

초기화 Interface

기화 판단 및 시행을 해준다.

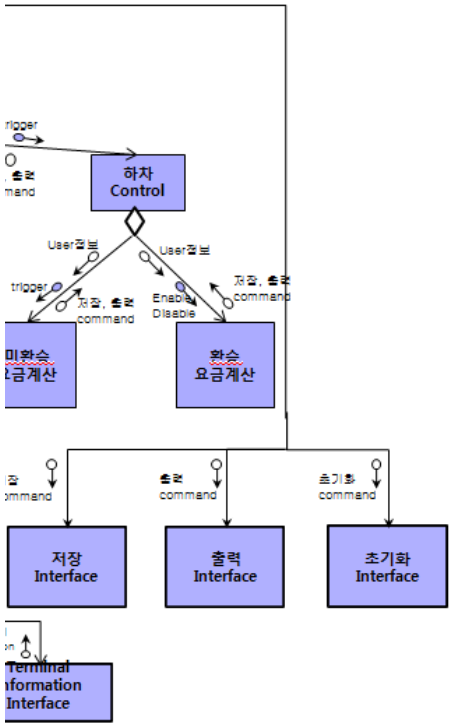
Code Analysis(7

Structured Chart – PTS (Advanced)

❖ 지하철

```
void initialize(){
FILE *f;
f = fopen("metro.txt", "w");
fputs("", f);
}
```

```
/*3분이면*/
if (secTime % 180 == 0 && secTime != 0){
//초기화
initialize();
initializer = 1;
printf("\n초기화!\n");

initTimer = time(NULL);
secWait = 0;
}
```



번호	1.6.2
이름	초기화
입력	Tick, Tr
출력	초기화
설명	3분주기

Code Analysis(7

Structured Chart – PTS (Advanced)

❖ 지하철

Main

```
if (initializer ==
{
//탑승/하차와지
//최초입력이면
userinfo.transp
userinfo.in_out
/*printf("처음입
//초기화했을때,
tempi = userin
if (tempi == IN
{
strcpy(userinfo
userinfo.balanc
strcpy(userinfo
/*tempi = 3;*/
//태그시간, 잔액
printf("초기화전
}
//초기화했을때
//이때는userint
//lasttagtime, t
else{
userinfo.in_out
userinfo.transp
strcpy(userinfo
userinfo.balanc
strcpy(userinfo
printf("초기화후
}
printf("초기화처
initializer = 0;
}
```

```
do{
currentinfo.transportation = METRO
printf("건대(b), 강남(c), 신림(d), 합정(e), 동역사(f)");
scanf("%c", &metrob);
fflush(stdin);

if (metrob[0] == 'b'){
printf("건대\n");
strcpy(currentinfo.terminalinfo, metrob);
/*if (ios == IN)
check_in();
else if (ios == OUT)
check_out();*/
check_in_out(&ios);
break
}
else if (metrob[0] == 'c'){
printf("강남\n");
strcpy(currentinfo.terminalinfo, metrob);
/*if (ios == IN)
check_in();
else if (ios == OUT)
check_out();*/
check_in_out(&ios);
break
}
else if (metrob[0] == 'd'){
printf("신림\n");
strcpy(currentinfo.terminalinfo, metrob);
/*if (ios == IN)
check_in();
else if (ios == OUT)
```

```
check_out();*/
check_in_out(&ios);
break
}
else if (metrob[0] == 'e'){
printf("합정\n");
strcpy(currentinfo.terminalinfo, metrob);
/*if (ios == IN)
check_in();
else if (ios == OUT)
check_out();*/
check_in_out(&ios);
break
}                                                    t != OUT) &&
else if (metrob[0] == 'f'){
printf("동역사\n");
strcpy(currentinfo.terminalinfo, metrob);
/*if (ios == IN)
check_in();
else if (ios == OUT)
check_out();*/
check_in_out(&ios);
break
}

else
{
}


} whil
//지하
```

```
void check_in_out(int *a)
{
    if (*a == IN)
        check_in();
    else if (*a == OUT)
        check_out();

}
```

subway_in.c

```
int check_in()
{
char* tempmin1;// userinfo
char* tempsec1;// userinfo
char* tempmin2;// currentinfo
char* tempsec2;// currentinfo
int check_trans;// 시간에따른환승여부판단
int totaltime1;
int totaltime2;
char temp1[10];// 시간잠시저장하기위한문자열배열!
char temp2[10];// 시간잠시저장하기위한문자열배열!
strcpy(temp1, currentinfo.lastTagTime);
strcpy(temp2, userinfo.lastTagTime);
tempmin1 = strtok(userinfo.lastTagTime, ":");
tempsec1 = strtok(NULL, ":");
totaltime1 = (atoi(tempmin1)) * 60 + atoi(tempsec1);
tempmin2 = strtok(currentinfo.lastTagTime, ":");
tempsec2 = strtok(NULL, ":");
totaltime2 = (atoi(tempmin2)) * 60 + atoi(tempsec2);
strcpy(currentinfo.lastTagTime, temp1);
strcpy(userinfo.lastTagTime, temp2);
check_trans = totaltime2 - totaltime1;
check = 0;
if (userinfo.in_out == IN && currentinf
{
if ((userinfo.terminalinfo[0] == 'a' && u
 currentinfo.terminalinfo[0] != 'a' &&cu
|| (userinfo.terminalinfo[0] != 'a' && us
currentinfo.terminalinfo[0] != 'a' && cu
{
```

currentinfo.balance = notcal_trans(); // 1750원OR 1650원(버스->지하철환승이1650원, 지하철->버스환승이1750원)
check = 2;
}
else if (userinfo.terminalinfo[0] != 'a' && userinfo.transportation == METRO && currentinfo.terminalinfo[0] != 'a' && currentinfo.transportation == METRO)
{
currentinfo.balance = notcal_nottrans(); // 1250원
check = 2;
}
else if (userinfo.terminalinfo[0] == 'a' && userinfo.transportation == BUS && currentinfo.terminalinfo[0] != 'a' && currentinfo.transportation == METRO)
{
currentinfo.balance = basic();// 기본료1050원: 버스승차후지하철승차
check = 2;
}
}
if (userinfo.in_out == OUT && currentinfo.in_out == IN)
{
if (userinfo.terminalinfo[0] == 'a' && userinfo.transportation == BUS && currentinfo.terminalinfo[0] != 'a' && currentinfo.transportation == METRO && check_trans <= 15)
{

currentinfo.terminalinfo[0] != 'a' && currentinfo.transportation == METRO && check_trans >15))
{
//basic();// 기본료1050원: 지하철하차후지하철승차, 최초탑승포함, 환승시간지난경우!
currentinfo.balance = basic(); // 1050원
check = 2;
}
}
}

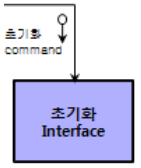
설명	실질적으 승여부, 식을판단

Code Analysis(7

Structured Chart – PTS (Advanced)

❖ 지하철


```
int notcal_trans() {// 미정산&환승(전예탄계버스)
int extra_fee_a = 600;// 버스->지하철환승후미정산추가요금
int extra_fee_b = 700;// 지하철->버스환승후미정산추가요금

if (userinfo.terminalinfo[0] != 'a' && userinfo.transportation == BUS &&
currentinfo.terminalinfo[0] != 'a' && currentinfo.transportation == METRO)
{
printf("%d원\n", BASIC_FEE + extra_fee_a);
return BASIC_FEE + extra_fee_a;
}
else if (userinfo.terminalinfo[0] == 'a' && userinfo.transportation == METRO &&
currentinfo.terminalinfo[0] != 'a' &&currentinfo.transportation == METRO)
{
printf("%d원\n", BASIC_FEE + extra_fee_b);
return BASIC_FEE + extra_fee_b;
}
}
```

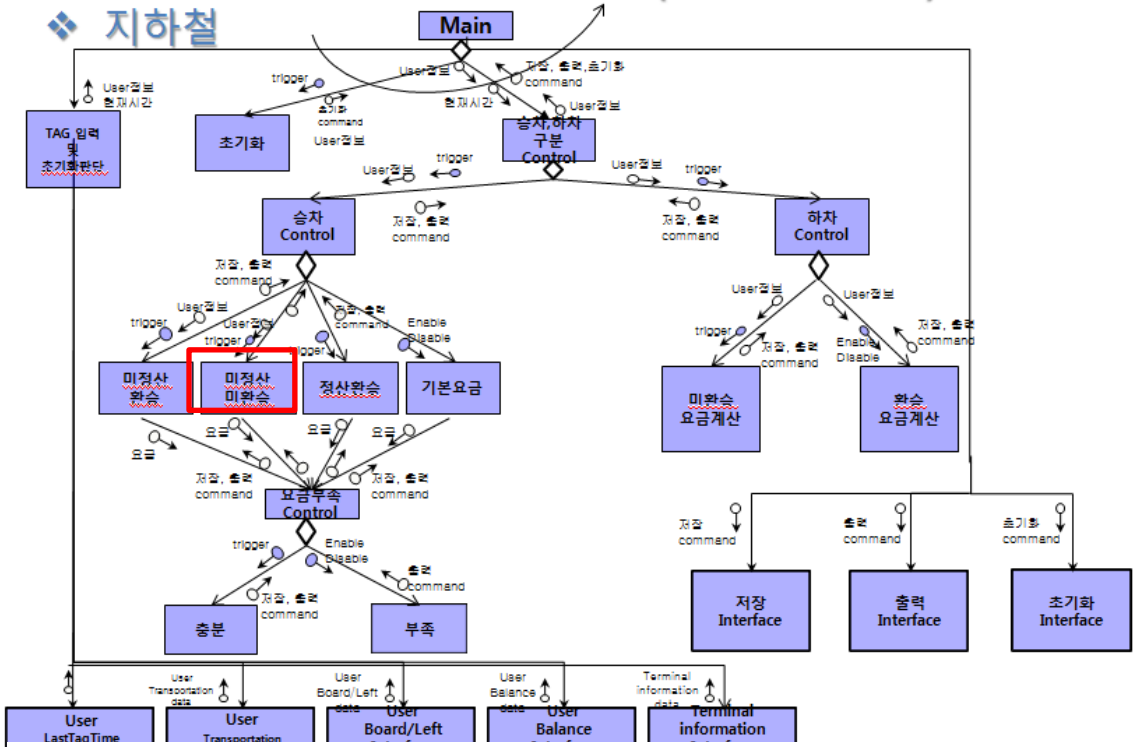


번호	
이름	
입력	
출력	
설명	

Code Analysis(7

Structured Chart – PTS (Advanced)

❖ 지하철



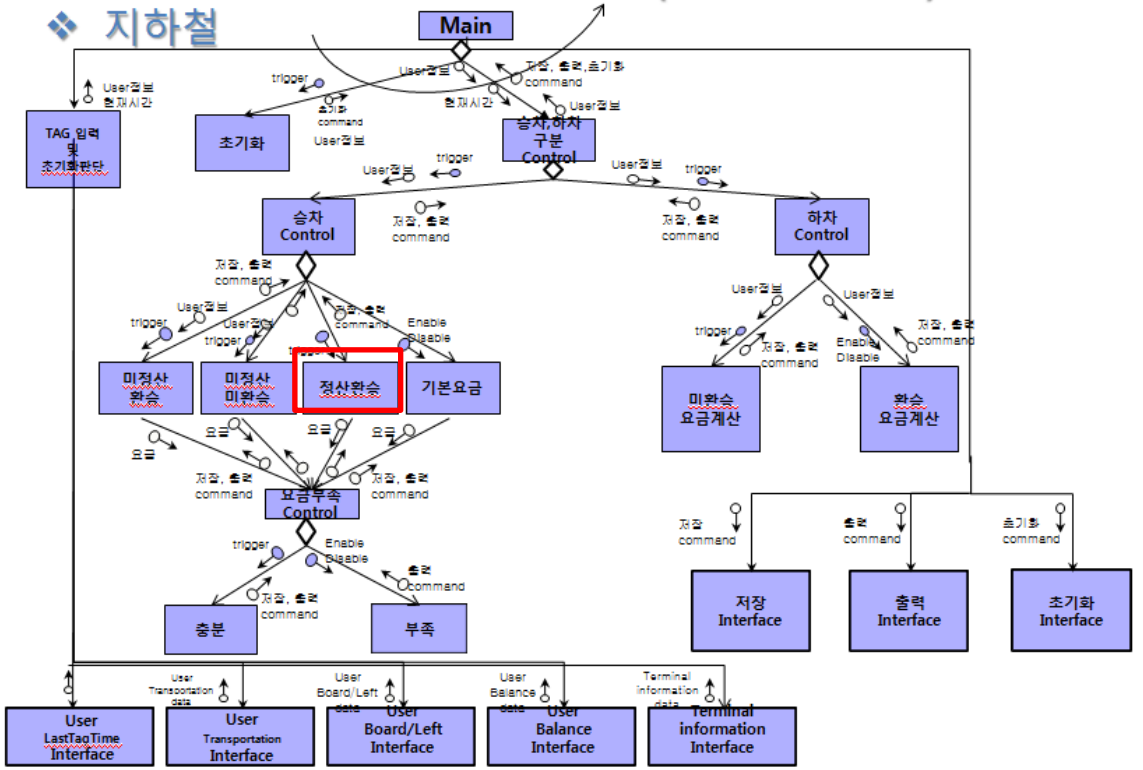
번호	지하철 2.2.3
이름	미정산 미환승
입력	tick, trigger
출력	요금
설명	사용자가 미정산을하고 환승을 안한 : 액들을 다음 컨트롤러로 전달해준다.

```
int notcal_nottrans() {// 미정산&미환승(전예탄계지하철)
int extra_fee = 200;
printf("%d원\n", BASIC_FEE + extra_fee);
return BASIC_FEE + extra_fee;
}
```

Code Analysis(7

Structured Chart – PTS (Advanced)

❖ 지하철



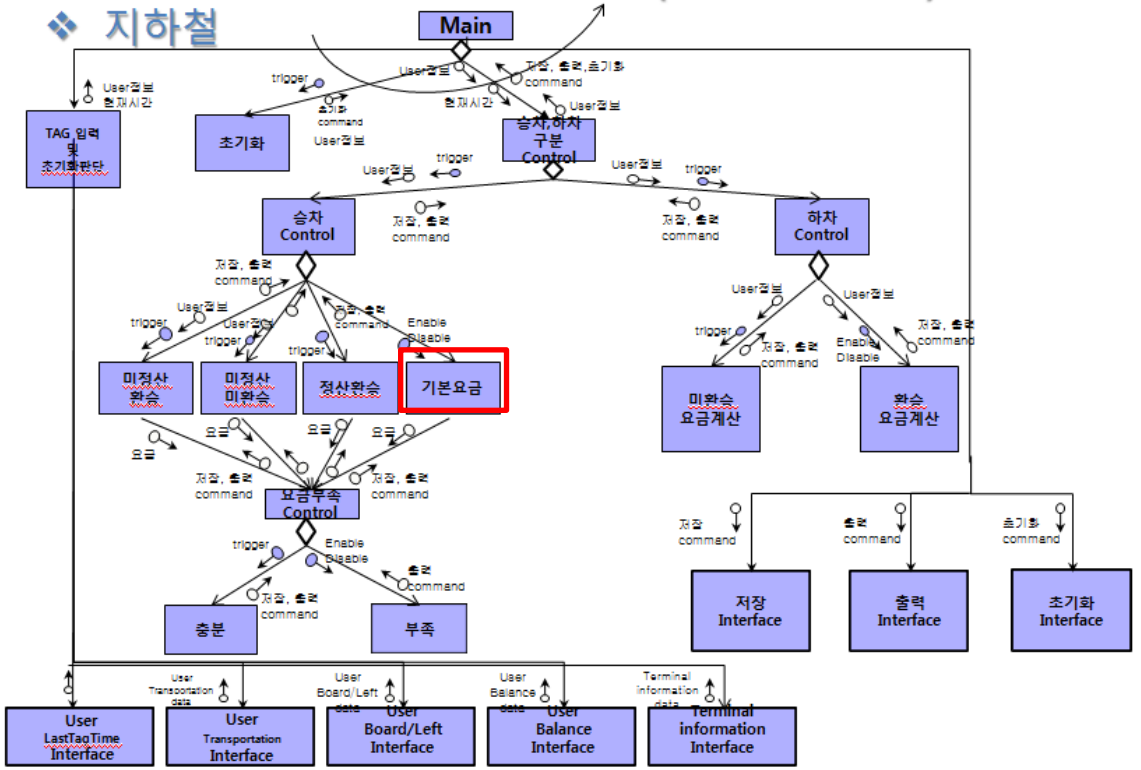
번호	지하철 2.2.4
이름	정산/환승
입력	User정보,tick, trigger
출력	요금
설명	사용자가 정산을하고 환승을 안한 경우의 금액을 계산하고, 최대금액을 겨 들을 다음 컨트롤러로 전달해준다.

```
int cal_trana() {// 정산&환승(전예탄계버스)
int zero = 0;
printf("%d원\n", zero);
return zero;
}
```

Code Analysis(7

Structured Chart – PTS (Advanced)

❖ 지하철



번호	지하철 2.2.5
이름	기본요금
입력	enable, disable
출력	요금
설명	사용자가 최초탑승이거나 이전 음 컨트롤러로 전달해준다.

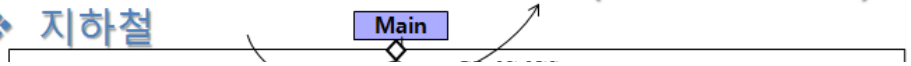
```
int basic() {// 처음타거나전에탄게버스인데내릴때안찍었을경우
printf("%d원\n", BASIC_FEE);
return BASIC_FEE;
}
```

Code Analysis(7

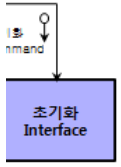
```
//잔액테스트가성공했을경우, 저장해준다
while (1)
{
if (ios == 1)//하차일때는요금부족판단안하므로!!
{
saveInfo();
break
}
else if (ios == 0)// 승차일때는요금부족확인하고!
{
balance_check();

if ((userinfo.balance >= money))
{
if (check == 2)
{
if (metrob[0] == 'b')
{
terminalCount[1]++;
sprintf(cterminalCount, "%d", terminalCount[1]);
strcpy(currentinfo.terminalinfo, metrob);
strcat(currentinfo.terminalinfo, cterminalCount);
}
else if (metrob[0] == 'c')
{
terminalCount[2]++;
sprintf(cterminalCount, "%d", terminalCount[2]);
strcpy(currentinfo.terminalinfo, metrob);
strcat(currentinfo.terminalinfo, cterminalCount);
}
else if (metrob[0] == 'd')
{
```

Structured Chart – PTS (Advanced)

❖ 지하철



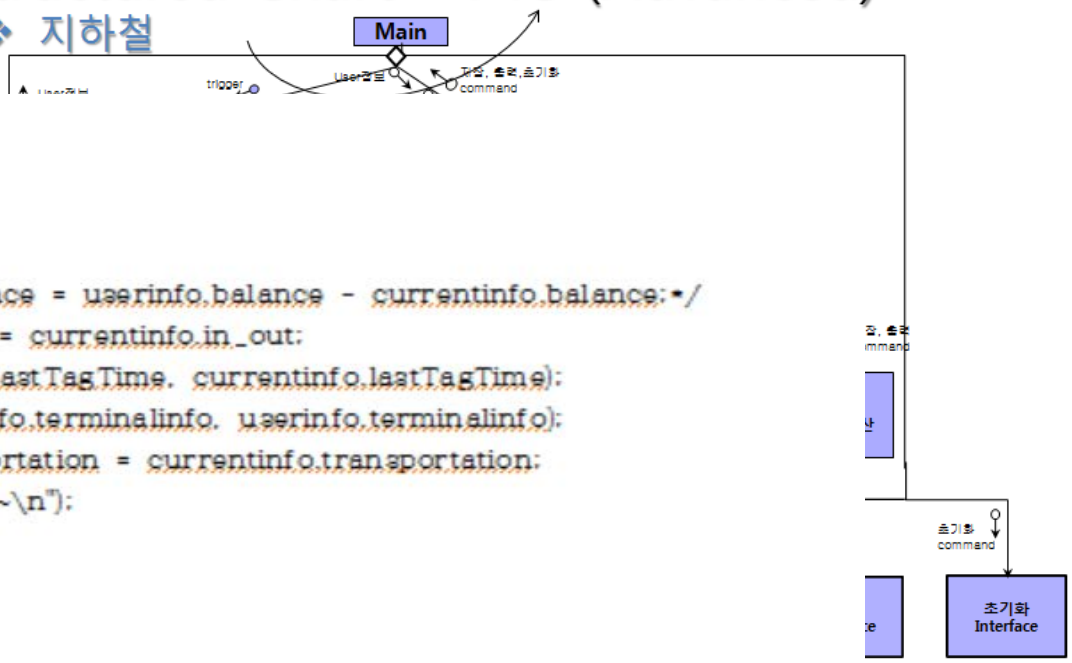

```
terminalCount[3]++;
sprintf(cterminalCount, "%d", terminalCount[3]);
strcpy(currentinfo.terminalinfo, metrob);
strcat(currentinfo.terminalinfo, cterminalCount);
}
else if (metrob[0] == 'e')
{
terminalCount[4]++;
sprintf(cterminalCount, "%d", terminalCount[4]);
strcpy(currentinfo.terminalinfo, metrob);
strcat(currentinfo.terminalinfo, cterminalCount);
}
else if (metrob[0] == 'f')
{
terminalCount[5]++;
sprintf(cterminalCount, "%d", terminalCount[5]);
strcpy(currentinfo.terminalinfo, metrob);
strcat(currentinfo.terminalinfo, cterminalCount);
}
}

strcpy(userinfo.terminalinfo, currentinfo.terminalinfo);
userinfo.balance = userinfo.balance - currentinfo.balance;
saveInfo();
}
break
}
else if ((userinfo.balance < money))
break
}
money = 0;
} }
```



Code Analysis(7

Structured Chart – PTS (Advanced)

❖ 지하철




```
else
{
money -= 600;
/*userinfo.balance = userinfo.balance - currentinfo.balance;*/
userinfo.in_out = currentinfo.in_out;
strcpy(userinfo.lastTagTime, currentinfo.lastTagTime);
strcpy(currentinfo.terminalinfo, userinfo.terminalinfo);
userinfo.transportation = currentinfo.transportation;
printf("승차고고~~\n");
}
}
```

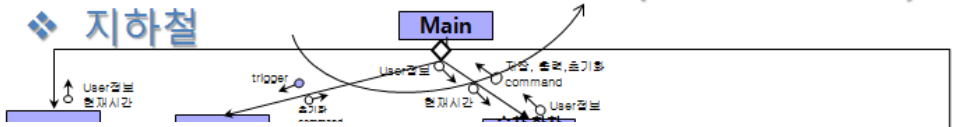
번호	지하철 2.:
이름	충분
입력	Trigger
출력	저장comm
설명	요금정보 간을 출력

```
else
{
money -= 200;
userinfo.in_out = currentinfo.in_out;
strcpy(userinfo.lastTagTime, currentinfo.lastTagTime);
userinfo.transportation = currentinfo.transportation;
printf("승차고고~~\n");
}
}
return 0
;}
```

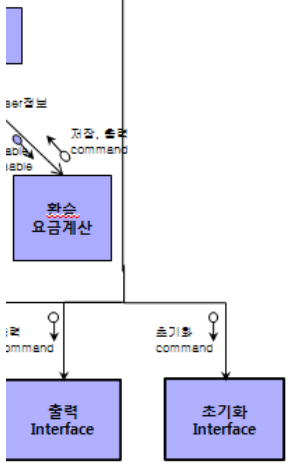
Code Analysis(7

Structured Chart – PTS (Advanced)

❖ 지하철




```
subway.c
int balance_check()
{
if (currentinfo.balance == BUS)// 환승인경우
{
money = currentinfo.balance + 600;
if (userinfo.balance < money)
printf("요금이부족해서승차하실수없습니다!!!\n");


else                                  // 환승이아닌경우
{
money = currentinfo.balance + 200;
if (userinfo.balance < money)
printf("요금이부족해서승차하실수없습니다!!!\n");
```

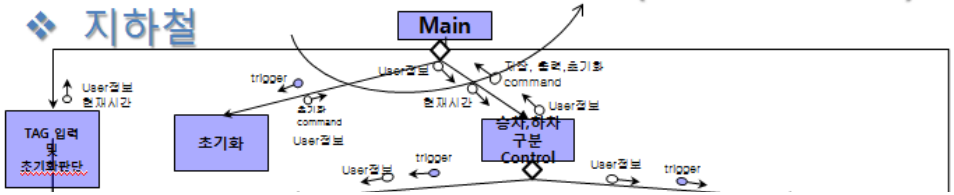


번호	
이름	
입력	
출력	
설명	금액이 부족한 경우 활성화 되며, 금액 부족 문구를 출력하는 명령을 전송한다.

Code Analysis(7

Structured Chart – PTS (Advanced)

❖ 지하철



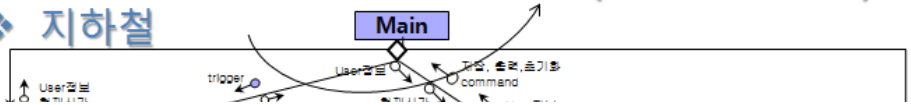
subway_out.c

```
int check_out()// 하차에서어떤요금을부과할지결정!
{
check = 0;
if (userinfo.in_out == IN && userinfo.tranaportation == METRO &&
currentinfo.in_out == OUT
&& currentinfo.tranaportation == METRO)
{
if (userinfo.terminalinfo[0] != 'a' && currentinfo.terminalinfo[0] != 'a')
{
currentinfo.balance = nottrana_cal(); // 0원, 200원둘중하나부과--> 0,1 정거장==0
원, 2정거장= 200
원/ 지하철승차후지하철하차한경우, 앞에환승기록없음
userinfo.balance = userinfo.balance - currentinfo.balance;
userinfo.in_out = currentinfo.in_out;
strcpy(userinfo.lastTagTime, currentinfo.lastTagTime);
strcpy(currentinfo.terminalinfo, userinfo.terminalinfo);
userinfo.transportation = currentinfo.transportation;
}
else if (userinfo.terminalinfo[0] == 'a' && currentinfo.terminalinfo[0] != 'a')
{
currentinfo.balance = trana_cal();// 300원, 600원둘중하나부과! --> 1정거장= 300원,
2정거장= 600원/ 버스에서지하철환승후)
userinfo.balance = userinfo.balance - currentinfo.balance;
userinfo.in_out = currentinfo.in_out;
strcpy(userinfo.lastTagTime, currentinfo.lastTagTime);
strcpy(currentinfo.terminalinfo, userinfo.terminalinfo);
userinfo.tranaportation = currentinfo.tranaportation;
}
}
return 0;
}
```

Code Analysis(7

Structured Chart – PTS (Advanced)

❖ 지하철

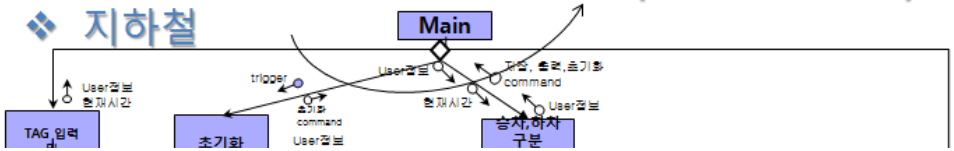



```
int nottrana_cal() {// 미환승(전예탄계지하철)요금계산
int station0 = 0;// 0 정거장이동
int station1 = 0;// 한정거장이동
int station2 = 200;// 두정거장이동
if ((userinfo.terminalinfo[0] == 'b' && currentinfo.terminalinfo[0] == 'b')
|| (userinfo.terminalinfo[0] == 'c' && currentinfo.terminalinfo[0] == 'c')
|| (userinfo.terminalinfo[0] == 'd' && currentinfo.terminalinfo[0] == 'd')
|| (userinfo.terminalinfo[0] == 'e' && currentinfo.terminalinfo[0] == 'e')
|| (userinfo.terminalinfo[0] == 'f' && currentinfo.terminalinfo[0] == 'f'))
{
printf("하차시추가요금: %d원\n", station0);

return station0;
}
else if ((userinfo.terminalinfo[0] == 'b' && (currentinfo.terminalinfo[0] == 'f'
|| currentinfo.terminalinfo[0] == 'c'))
|| (userinfo.terminalinfo[0] == 'c' && (currentinfo.terminalinfo[0] == 'b'
|| currentinfo.terminalinfo[0] == 'd'))
|| (userinfo.terminalinfo[0] == 'd' && (currentinfo.terminalinfo[0] == 'c'
|| currentinfo.terminalinfo[0] == 'e'))
|| (userinfo.terminalinfo[0] == 'e' && (currentinfo.terminalinfo[0] == 'd'
|| currentinfo.terminalinfo[0] == 'f'))
|| (userinfo.terminalinfo[0] == 'f' && (currentinfo.terminalinfo[0] == 'e'
|| currentinfo.terminalinfo[0] == 'b')))
{
printf("하차시추가요금: %d원\n", station1);
return station1;
}
else if ((userinfo.terminalinfo[0] == 'b' && (currentinfo.terminalinfo[0] == 'd'
|| currentinfo.terminalinfo[0] == 'e'))
|| (userinfo.terminalinfo[0] == 'c' && (currentinfo.terminalinfo[0] == 'e'
|| currentinfo.terminalinfo[0] == 'f'))
|| (userinfo.terminalinfo[0] == 'd' && (currentinfo.terminalinfo[0] == 'f'
|| currentinfo.terminalinfo[0] == 'b'))
|| (userinfo.terminalinfo[0] == 'e' && (currentinfo.terminalinfo[0] == 'b'
|| currentinfo.terminalinfo[0] == 'c'))
|| (userinfo.terminalinfo[0] == 'f' && (currentinfo.terminalinfo[0] == 'c'
|| currentinfo.terminalinfo[0] == 'd')))
{
printf("하차시추가요금: %d원\n", station2);
return station2;
}
}
```

Code Analysis(7

Structured Chart – PTS (Advanced)

❖ 지하철




```
int trans_cal() {// 환승(전예탄계버스)요금계산
int station0 = 0;// 0 정거장이동
int station1 = 300;// 한정거장이동
int station2 = 600;// 두정거장이동
if
((c == 'b' && currentinfo.terminalinfo[0] == 'b')
|| (c == 'c' && currentinfo.terminalinfo[0] == 'c')
|| (c == 'd' && currentinfo.terminalinfo[0] == 'd')
|| (c == 'e' && currentinfo.terminalinfo[0] == 'e')
|| (c == 'f' && currentinfo.terminalinfo[0] == 'f'))
{
printf("하차시추가요금: %d원\n", station0);
return station0;
}
else if
((c == 'b' && (currentinfo.terminalinfo[0] == 'c'
|| currentinfo.terminalinfo[0] == 'f'))
|| (c == 'c' && (currentinfo.terminalinfo[0] == 'd'
 || currentinfo.terminalinfo[0] == 'b'))
|| (c == 'd' && (currentinfo.terminalinfo[0] == 'e'
|| currentinfo.terminalinfo[0] == 'c'))
|| (c == 'e' && (currentinfo.terminalinfo[0] == 'f'
 || currentinfo.terminalinfo[0] == 'd'))
|| (c == 'f' && (currentinfo.terminalinfo[0] == 'b'
|| currentinfo.terminalinfo[0] == 'e')))
{
printf("하차시추가요금: %d원\n", station1);
return station1;
}
else if
((c == 'b' && (currentinfo.terminalinfo[0] == 'd' || currentinfo.terminalinfo[0] ==
'e'))
|| (c == 'c' && (currentinfo.terminalinfo[0] == 'e' || currentinfo.terminalinfo[0] ==
'f'))
|| (c == 'd' && (currentinfo.terminalinfo[0] == 'f' || currentinfo.terminalinfo[0] ==
'b'))
|| (c == 'e' && (currentinfo.terminalinfo[0] == 'b' || currentinfo.terminalinfo[0] ==
'c'))
|| (c == 'f' && (currentinfo.terminalinfo[0] == 'c' || currentinfo.terminalinfo[0] ==
'd')))
{
printf("하차시추가요금: %d원\n", station2);
return station2;
}
}
```

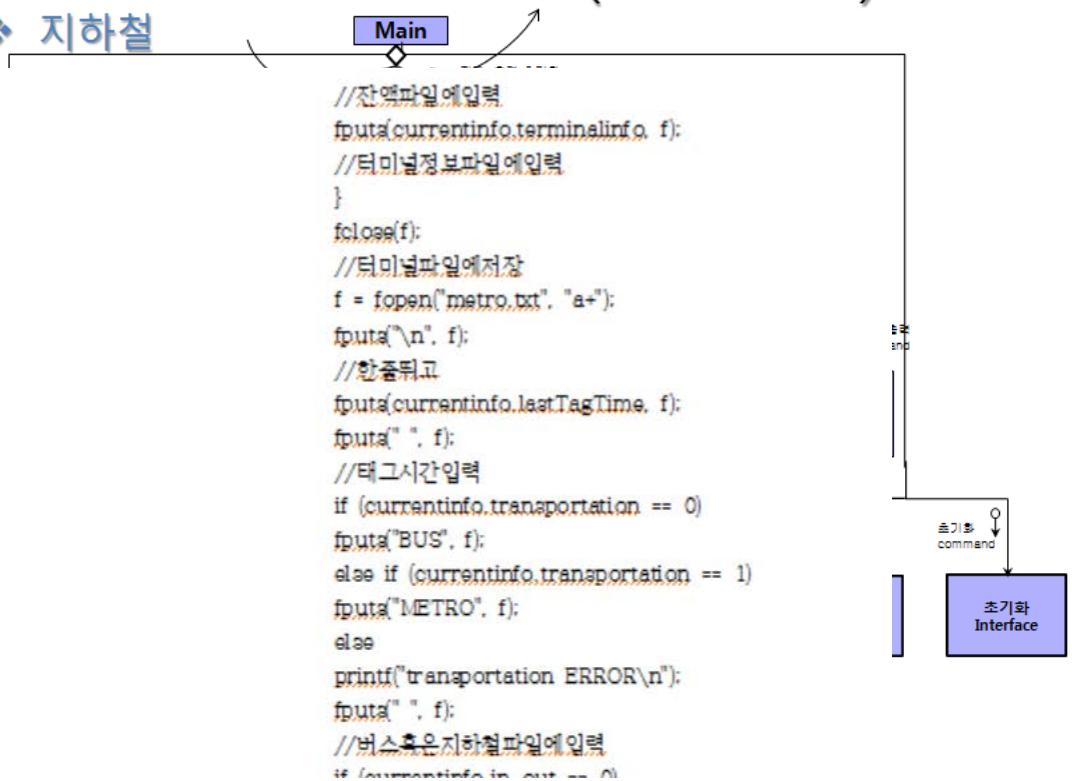
	산을 출력한다.

Code Analysis(7

```
void saveInfo(){
FILE *f;
char cbalance[10];//잔액char 배열로할당하기위한값
f = fopen("C:\\수민\\test\\user.txt", "a");
if (f == NULL){ perror("사용자파일이없습니다"); }
else{
fputs("\n", f);
//한줄뛰고
fputs(currentinfo.lastTagTime, f);
fputs(" ", f);
//태그시간입력
if (currentinfo.transportation == 0)
fputs("BUS", f);
else if (currentinfo.transportation == 1)
fputs("METRO", f);
else
printf("transportation ERROR\n");
fputs(" ", f);
//버스혹은지하철파일에입력
if (currentinfo.in_out == 0)
fputs("IN", f);
else if (currentinfo.in_out == 1)
fputs("OUT", f);
else
printf("in_out ERROR\n");
fputs(" ", f);
//탑승하차파일에입력
/*realBalance = userinfo.balance - currentinfo.balance;*/
sprintf(cbalance, "%d", userinfo.balance);
// c배열에잔액입력함수가캐릭터배열로입력해야되거든
fputs(cbalance, f);
fputs(" ", f);
```

Structured Chart – PTS (Advanced)

❖ 지하철




```
//잔액파일에입력
fputs(currentinfo.terminalinfo, f);
//터미널정보파일에입력
}
fclose(f);
//터미널파일에저장
f = fopen("metro.txt", "a+");
fputs("\n", f);
//한줄뛰고
fputs(currentinfo.lastTagTime, f);
fputs(" ", f);
//태그시간입력
if (currentinfo.transportation == 0)
fputs("BUS", f);
else if (currentinfo.transportation == 1)
fputs("METRO", f);
else
printf("transportation ERROR\n");
fputs(" ", f);
//버스혹은지하철파일에입력
```

```
//잔액파일에입력
fputs(currentinfo.terminalinfo, f);
//터미널정보파일에입력

fclose(f);
}
```

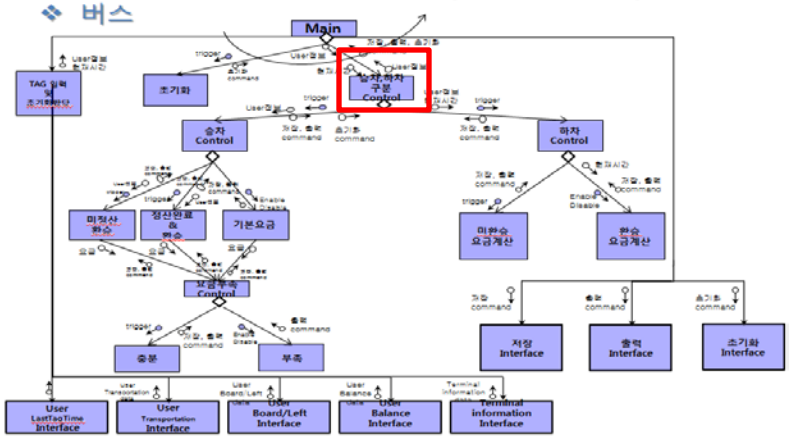
Code Analysis(ㅂㅣ스)

Code Analysis(버스)

번호	버스 2.1.1
이름	승차, 하차 구분control
입력	User 정보, 현재시간
출력	trigger
설명	승객의 승차, 하차를 구분 짓는 컨트롤로 사용자 정보와 현재시간을 입력 받아 승차인지 하차인지 결정 짓는다.

Structured Chart – PTS (Advanced)

❖ 버스



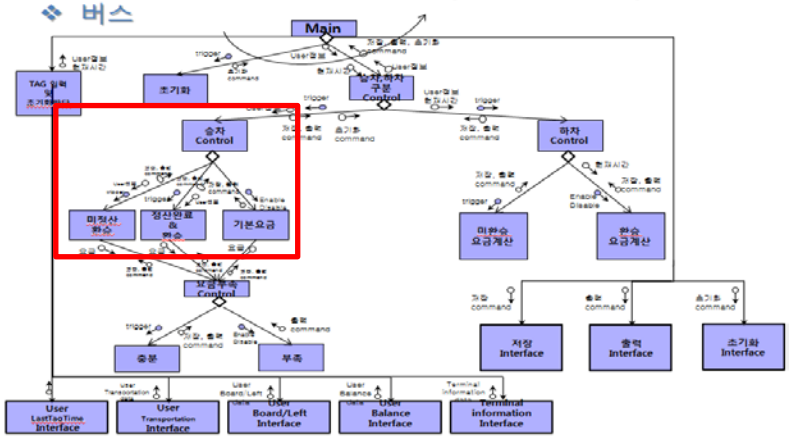

```
int busin_outControl() //버스 승하차 컨트롤
{
        if (currentinfo.in_out == IN)
        {
                return busChargeShortageControl(busBoardControl());
        }
        else{
                busLeftControl();
                strcpy(currentinfo.terminalinfo, userinfo.terminalinfo);
                return 1;
        }
}
```

Code Analysis(버스)

번호	버스 2.2.4
이름	미정산/환승
입력	tick, trigger, User 정보
출력	요금
설명	사용자가 미정산을하고 환승을 안한 경우의 금액을 계산하고, 최대금액을 계산한 후 해당 금액들을 다음 컨트롤러로 전달해준다.

Structured Chart – PTS (Advanced)

❖ 버스



번호	
이름	
입력	
출력	
설명	

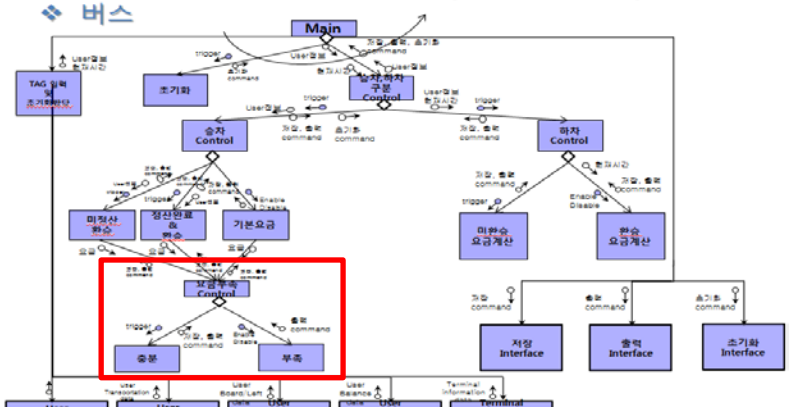
```
int busBoardControl() //버스 승차컨트롤
{
	int result = 10;
	char* tempmin1;
	char* tempsec1;
	char* tempmin2;
	char* tempsec2;
	int check_trans;   // 시간에 따른 환승 여부 판단
	int totaltime1;
	int totaltime2;
	char temp[10];          // 시간 잠시 저장하기 위한 문자열배열!
	char temp2[10];
	int cccc;

	strcpy(temp, currentinfo.lastTagTime);
	strcpy(temp2, userinfo.lastTagTime);
	tempmin1 = strtok(userinfo.lastTagTime, ":");
	tempsec1 = strtok(NULL, ":");
	totaltime1 = (atoi(tempmin1)) * 60 + atoi(tempsec1);
	tempmin2 = strtok(currentinfo.lastTagTime, ":");
	tempsec2 = strtok(NULL, ":");
	totaltime2 = (atoi(tempmin2)) * 60 + atoi(tempsec2);
	strcpy(currentinfo.lastTagTime, temp);
	strcpy(userinfo.lastTagTime, temp2);

	if (userinfo.in_out == OUT && userinfo.transportation == METRO &&
		15>totaltime2-totaltime1) //지하철 찍고 하차후 버스 승차
	{
		result = 1;//최대 요금 500원 1번
		return result;
	}

	else if ((userinfo.transportation == BUS && userinfo.in_out == OUT) ||
		(userinfo.transportation == BUS && userinfo.terminalinfo[0] == 'a')
		|| (userinfo.in_out == OUT && userinfo.transportation == METRO
		&& userinfo.terminalinfo[0] != 'a' && 15<totaltime2 - totaltime1))
	{
		result = 2;//기본요금 1050원 2번
		return result;
	}
	else if (userinfo.in_out == IN)
	{
		if (userinfo.terminalinfo[0] == 'a' && userinfo.transportation ==
			METRO) //버스->지하철 하차시 노 태그
		{
			result = 3;//최대 요금 1650원 3번
			return result;
		}

		else if (userinfo.terminalinfo[0] != 'a' && userinfo.transportation
			== BUS) // 지하철->버스 하차시 노 태그
		{
			result = 4;//최대 요금 1750원 4번
			return result;
		}

		else if (userinfo.terminalinfo[0] != 'a' && userinfo.transportation
			== METRO) //지하철 하차시 노 태그
		{
			result = 5;//최대 요금 1250원 5번
			return result;
		}
	}
	else{
		return result;
	}
}
```

Code Analysis(버스)

번호	버스 2.2.5
이름	요금부족 Control
입력	요금, User정보
출력	enable, disable, trigger
설명	전달받은 금액을 유저의 잔액과 비교해서 요금부족 여부를 판단하여 각각 트리거를 전달해준다.

Structured Chart – PTS (Advanced)

버스




```
int busChargeShortageControl(int result) //승차시에 요금부족 컨트롤
{
	char oterminalCount[4];
	int charge = 3;

	switch (result)
	{
	case 1:
		if (userinfo.balance<500)
		{
			charge = 0;
		}
		else//충분할 시에 계산
		{
			charge = 1;
			currentinfo.balance = 0;
			strcpy(currentinfo.terminalinfo, userinfo.terminalinfo);
		}
		return charge;
		break;
	case 2:
		if (userinfo.balance<1050)
		{
			charge = 0;
		}
		else//충분할 시에 계산
		{
			charge = 1;
			currentinfo.balance = 1050;
			terminalCount++;
			sprintf(oterminalCount, "%d", terminalCount);
			strcpy(currentinfo.terminalinfo, "a_");
			strcat(currentinfo.terminalinfo, oterminalCount);
		}
		return charge;
		break;
	case 3:
		if (userinfo.balance<1650)
		{
			charge = 0;
		}
		else//충분할 시에 계산
		{
			charge = 1;
			currentinfo.balance = 1650;
			terminalCount++;
			sprintf(oterminalCount, "%d", terminalCount);
			strcpy(currentinfo.terminalinfo, "a_");
			strcat(currentinfo.terminalinfo, oterminalCount);
		}
		return charge;
		break;
	case 4:
		if (userinfo.balance<1750)
		{
			charge = 0;
		}
		else//충분할 시에 계산
		{
			charge = 1;
			currentinfo.balance = 1750;
			terminalCount++;
			sprintf(oterminalCount, "%d", terminalCount);
			strcpy(currentinfo.terminalinfo, "a_");
			strcat(currentinfo.terminalinfo, oterminalCount);
		}
		return charge;
		break;
	case 5:
		if (userinfo.balance<1250)
```

번호	
이름	
입력	
출력	
설명	

번호	
이름	
입력	
출력	
설명	

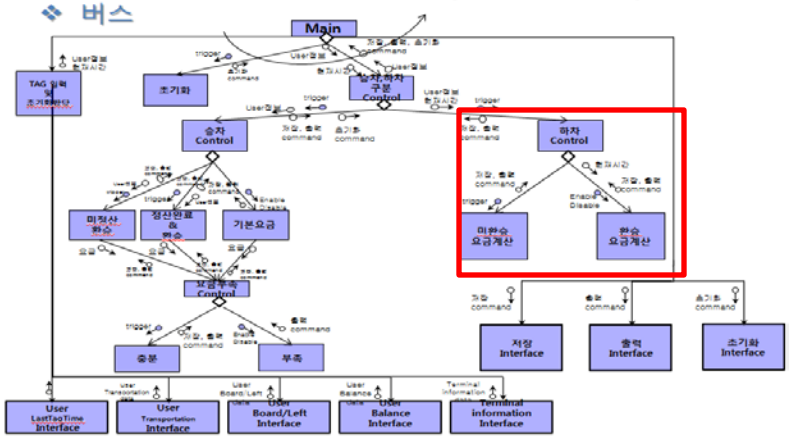
KU KONKUK UNIVERSITY

Code Analysis(버스)

번호	버스 2.3.1
이름	하차 Control
입력	trigger
출력	Trigger, Enable, Disable
설명	사용자가 하차태그를 하였을 시 환승을 하였는지 안 하였는지의 여부에 대한 트리거를 다음 프로세스에 전송해준다.

Structured Chart – PTS (Advanced)

❖ 버스



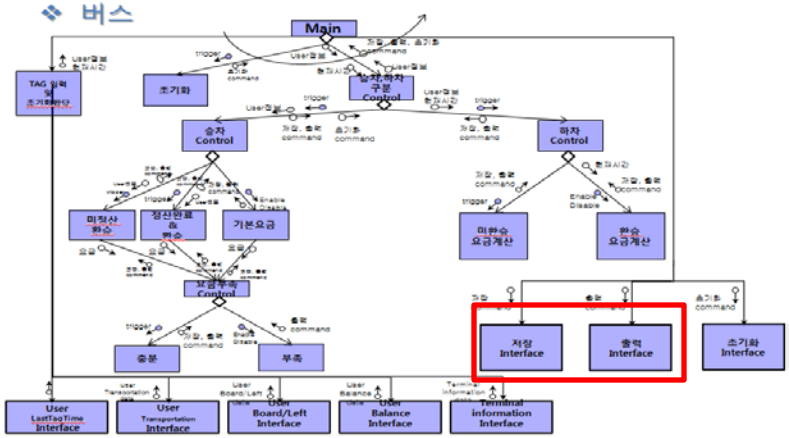

```
void busLeftControl() //버스 하차컨트롤
{
        int result = 10;
        char* tempmin1;
        char* tempsec1;
        char* tempmin2;
        char* tempsec2;
        int check_trans;   // 시간에 따른 환승 여부 판단
        int totaltime1;
        int totaltime2;
        char temp[10];          // 시간 잠시 저장하기 위한 문자열배열!
        char temp2[10];

        strcpy(temp, currentinfo.lastTagTime);
        strcpy(temp2, userinfo.lastTagTime);
        tempmin1 = strtok(userinfo.lastTagTime, ":");
        tempsec1 = strtok(NULL, ":");
        totaltime1 = (atoi(tempmin1)) * 60 + atoi(tempsec1);
        tempmin2 = strtok(currentinfo.lastTagTime, ":");
        tempsec2 = strtok(NULL, ":");
        totaltime2 = (atoi(tempmin2)) * 60 + atoi(tempsec2);
        strcpy(currentinfo.lastTagTime, temp);
        strcpy(userinfo.lastTagTime, temp2);

        if ((30>totaltime2 - totaltime1) || (userinfo.in_out == IN &&
                userinfo.terminalinfo[0] == 'a'))
        {
                currentinfo.balance = 0;
        }

        else if (60>totaltime2 - totaltime1)
        {
                currentinfo.balance = 100;
        }

        else if (90>totaltime2 - totaltime1)
        {
                currentinfo.balance = 200;
        }

        else if (120>totaltime2 - totaltime1)
        {
                currentinfo.balance = 300;
        }

        else if (150>totaltime2 - totaltime1)
        {
                currentinfo.balance = 400;
        }

        else
        {
                currentinfo.balance = 500;
        }
}
```

Code Analysis(버스)

```
if (feeb == 0){
        printf("요금이 부족합니다\n");
}
else if (feeb == 1){

        saveInfo();
        printf("시간 : %s \n", currentinfo.lastTagTime);
        printf("금액 : %d \n", currentinfo.balance);
        printf("잔액 : %d \n", userinfo.balance -
                currentinfo.balance);
}
printf("끝나고 user inout%d\n", userinfo.in_out);
```

Structured Chart – PTS (Advanced)

버스




```
void saveInfo(){
        FILE *f;
        char cbalance[10];          //잔액 char 배열로 할당하기 위한 값

        f = fopen("C:\\user.txt", "a");
        if (f == NULL)    { perror("사용자파일이 없습니다"); }

        else{
                fputs("\n", f);
                //한줄띄고

                fputs(currentinfo.lastTagTime, f);
                fputs(" ", f);
                //태그시간 입력


                if (currentinfo.transportation == 0)
                        fputs("BUS", f);
                else if (currentinfo.transportation == 1)
                        fputs("METRO", f);
                else
                        printf("transportation ERROR\n");
                fputs(" ", f);
                //버스 혹은 지하철 파일에 입력

                if (currentinfo.in_out == 0)
                        fputs("IN", f);
                else if (currentinfo.in_out == 1)
                        fputs("OUT", f);
                else
                        printf("in_out ERROR\n");
                fputs(" ", f);
                //탑승 하차 파일에 입력

                sprintf(cbalance, "%d", userinfo.balance - currentinfo.balance);
                // c배열에 잔액 입력 함수가 캐릭터 배열로 입력해야되서든
                fputs(cbalance, f);
                fputs(" ", f);
                //잔액 파일에 입력

                fputs(currentinfo.terminalinfo, f);
                //터미널정보 파일에 입력
        }

        folose(f);


        //터미널 파일에 저장
        f = fopen("bus.txt", "a+");

        fputs("\n", f);
        //한줄띄고

        fputs(currentinfo.lastTagTime, f);
        fputs(" ", f);
        //태그시간 입력


        if (currentinfo.transportation == 0)
                fputs("BUS", f);
        else if (currentinfo.transportation == 1)
                fputs("METRO", f);
        else
                printf("transportation ERROR\n");
        fputs(" ", f);
        //버스 혹은 지하철 파일에 입력

        if (currentinfo.in_out == 0)
                fputs("IN", f);
```

Code Analysis(정산)

Code Analysis(

Structured Chart – PTS (Advanced)

❖ 정산

```
void getInfo(struct result* rr){
struct termInfo infos[MAX_SIZE];
char termInput[MAX
int j;
char tempchar[MAX
char endFile[4];
char* line_p;
int ti;
int tj = 0;
int tk;
char tempc[20];
int sec1;
int sec2;
int size = 0;//갈은터

FILE *f;
leng = 0;
//지하철에서
f = fopen("metro.txt"

if (f == NULL) perro
//지하철정보들을받아
else {
while (fgets(termInp
때까지
```

```
if (leng >= MAX_SIZE){
printf("FULL\n");
}

printf("IN OUT 이아닙니다\n");

strcpy(tempchar, strtok(NULL, " "));//요금
infos[leng].balance = atoi(tempchar);

strcpy(tempchar, strtok(NULL, " "));//터미널
if ((line_p = strchr(tempchar, '\n')) != NULL)*line_p = '\0'
strcpy(infos[leng].terminalinfo, tempchar);

leng++;
}
}
fclose(f);
}
```

Code Analysis(

Structured Chart – PTS (Advanced)

❖ 정산

```
void getInfo(struct result* rr){
struct termInfo infos[MAX_SIZE];
char termInput[MAX_INPUT_SIZE];
int j;
char tempchar[MAX_INPUT_SIZE];
char endFile[4];
char* line_p;
int ti;
int tj = 0;
int tk;
char tempc[20];
int sec1;
int sec2;
int size = 0;//같은터미널정보의개수

FILE *f;
leng = 0;
f = fopen("bus.txt", "r");
if (f == NULL) perror("버스파일위치를
//지하철정보들을받아다입력
else {
while (fgets(termInput, MAX_INPUT_
때까지
if (leng >= MAX_SIZE)
printf("터미널정보최대치\n");
```

```
else{
strcpy(tempchar, strtok(termInput, " "));//태그시간
strcpy(infos[leng].lastTagTime, tempchar);

strcpy(tempchar, strtok(NULL, " "));//교통수단
if (strcmp(tempchar, "BUS") == 0){
infos[leng].transportation = BUS
}

infos[leng].balance = atoi(tempchar);

strcpy(tempchar, strtok(NULL, " "));//터미널
if ((line_p = strchr(tempchar, '\n')) != NULL)*line_p = '\0'
strcpy(infos[leng].terminalinfo, tempchar);

leng++;
}
}
fclose(f);
}
```

번 이 입 출 설

Code Analysis(Structured Chart – PTS (Advanced)

❖ 정산

```
        strcpy(rr[rleng].lastTagTime, infos[tk].lastTagTime);
        strcpy(rr[rleng].terminalinfo, infos[tk].terminalinfo);
        rleng++;
        }

        else{
        for (ti = tk - size; ti <= tk; ti += 2){
        //홀수번째원소이면다저장해준다.
        if (ti == leng - 2){
        rr[rleng].fee = infos[ti].balance;
        rr[rleng].transportation = infos[ti].transportation;
        strcpy(rr[rleng].lastTagTime, infos[ti].lastTagTime);
        strcpy(rr[rleng].terminalinfo, infos[ti].terminalinfo);
        rleng++;
        }
        else{
        rr[rleng].fee = infos[ti].balance;
        rr[rleng].transportation = infos[ti].transportation;
        strcpy(rr[rleng].lastTagTime, infos[ti].lastTagTime);
        strcpy(rr[rleng].terminalinfo, infos[ti].terminalinfo);
        //짝수번째는금액만저장해준다
        rr[rleng].fee += infos[ti + 1].balance;
        rleng++;
        }
        }
        }
        size = 0;
        }
        }

        for (j = 0; j<rleng; j++){
        printf("\n%d번째: lt = %s\n", j, rr[j].lastTagTime);
        printf("%d번째: tr = %d\n", j, rr[j].transportation);
        printf("%d번째: fee = %d\n", j, rr[j].fee);
        printf("%d번째: fee = %s\n", j, rr[j].terminalinfo);
        }
        }
```

Code Analysis(Structured Chart – PTS (Advanced)

❖ 정산

```
void adjustment(struct result* rt){
int size = 0;
int ai;
int aj;
int ri;
int rj = 0;
int total = 0;
int temp = 0;
int fee = 0;
int j;

for (ai = 0; ai<rleng - 1; ai++){
if ((strcmp(rt[ai].terminalinfo, rt[
//터미널정보가같은경우에size 늘려
size++;
printf("SAME\n");
if (ai == rleng - 2){
printf("SIZE:%d  ai:%d\n", size, a
for (ri = ai - size + 1; ri <= ai +
for (rj = ai + 1; rj >= ri; rj--){
total += rt[rj].fee;
}
fee += rt[ri].fee;
}
printf("total:%d\n", total);
for (ri = ai - size + 1; ri <= ai + 1; ri++){
for (rj = ai + 1; rj >= ri; rj--){

}
printf("total:%d\n", total);
for (ri = ai - size; ri <= ai; ri++){
for (rj = ri; rj <= ai; rj++){
temp += rt[rj].fee;
}
rt[ri].fee = round(((double)temp / (double)total)*(double)fee);
}
}
temp = 0;
total = 0;
size = 0;
fee = 0;
}
}

for (j = 0; j<rleng; j++){
printf("\n%d번째: lt = %s\n", j, rt[j].lastTagTime);
printf("%d번째: tr = %d\n", j, rt[j].transportation);
printf("%d번째: fee = %d\n", j, rt[j].fee);
printf("%d번째: fee = %s\n", j, rt[j].terminalinfo);
}
}
fee += rt[ri].fee;
```

p / (double)total)*(double)fee);

위까지정산시행

는그냥둔다

, ai);

i++){

Unit Test

지하철test(초기화 후 첫 탑승)

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:2 METRO IN 8950 b_1
0:4 METRO OUT 8750 b_1
```

C:₩Windows₩system32₩cmd.exe

```
태그할 경우 t 입력:
00:00
00:01
00:02
t
==================
USER INFORMATION
==================
transport : n
inout:n
balance: 10000
초기화 후 최초탑승
초기화처리 후 inout 1
탑승(0) 하차(1) : 0
IN
건대(b), 강남(c), 신림(d), 합정(e), 동역사(f)b
건대
1050원
승차 고고~~
태그할 경우 t 입력:
00:02
00:03
00:04
t
==================
USER INFORMATION
==================
lasttagtime : 0:2
transport : METRO
inout:IN
balance: 8950
terminal:b_1
탑승(0) 하차(1) : 1
OUT
건대(b), 강남(c), 신림(d), 합정(e), 동역사(f)e
합정
하차시 추가요금 : 200원
```

KU KONKUK UNIVERSITY

지하철test(버스 태그 하차 후 지하철 탑승)

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:2 METRO IN 8950 b_1
0:4 METRO OUT 8750 b_1
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:2 METRO IN 8950 b_1
0:4 BUS OUT 8750 a_1
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:2 METRO IN 8950 b_1
0:4 BUS OUT 8750 a_1
0:8 METRO IN 8750 a_1
0:13 METRO OUT 8150 a_1
```

```
태그할 경우 t 입력:
00:04
00:05
00:06
00:07
00:08
t
==================
USER INFORMATION
==================
lasttagtime : 0:4
transport : BUS
inout:OUT
balance: 8750
terminal:a_1
탑승(0) 하차(1) : 0
IN
건대(b), 강남(c), 신림(d), 합정(e), 동역사(f)c
강남
0원
승차 고고~~


태그할 경우 t 입력:
00:08
00:09
00:10
00:11
00:12
00:13
t
==================
USER INFORMATION
==================
lasttagtime : 0:8
transport : METRO
inout:IN
balance: 8750
terminal:a_1
탑승(0) 하차(1) : 1
OUT
건대(b), 강남(c), 신림(d), 합정(e), 동역사(f)f
동역사
하차시 추가요금 : 600원
```

KU KONKUK UNIVERSITY

지하철test(지하철 태그 하차 후 지하철 탑승)

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:2 METRO IN 8950 b_1
0:4 BUS OUT 8750 a_1
0:8 METRO IN 8750 a_1
0:13 METRO OUT 8150 a_1
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:2 METRO IN 8950 b_1
0:4 BUS OUT 8750 a_1
0:8 METRO IN 8750 a_1
0:13 METRO OUT 8150 a_1
0:17 METRO IN 7100 d_1
0:20 METRO OUT 7100 d_1
```

```
태그할 경우 t 입력:
00:13
00:14
00:15
00:16
00:17
t
==================
USER INFORMATION
==================
lasttagtime : 0:13
transport : METRO
inout:OUT
balance: 8150
terminal:a_1
탑승(0) 하차(1) : 0
IN
건대(b), 강남(c), 신림(d), 합정(e), 동역사(f)d
신림
1050원
승차 고고~~
태그할 경우 t 입력:
00:17
00:18
00:19
00:20
t
==================
USER INFORMATION
==================
lasttagtime : 0:17
transport : METRO
inout:IN
balance: 7100
terminal:d_1
탑승(0) 하차(1) : 1
OUT
건대(b), 강남(c), 신림(d), 합정(e), 동역사(f)d
신림
하차시 추가요금 : 0원
```

KU KONKUK UNIVERSITY

지하철test(지하철 태그x 하차 후 지하철 탑승)

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:2 METRO IN 8950 b_1
0:4 BUS OUT 8750 a_1
0:8 METRO IN 8750 a_1
0:13 METRO OUT 8150 a_1
0:17 METRO IN 7100 d_1
0:20 METRO OUT 7100 d_1
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:2 METRO IN 8950 b_1
0:4 BUS OUT 8750 a_1
0:8 METRO IN 8750 a_1
0:13 METRO OUT 8150 a_1
0:17 METRO IN 7100 d_1
0:20 METRO IN 7100 c_1
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:2 METRO IN 8950 b_1
0:4 BUS OUT 8750 a_1
0:8 METRO IN 8750 a_1
0:13 METRO OUT 8150 a_1
0:17 METRO IN 7100 d_1
0:20 METRO IN 7100 c_1
0:24 METRO IN 5850 d_2
0:28 METRO OUT 5650 d_2
```

```
태그할 경우 t 입력:
00:20
00:21
00:22
00:23
00:24
t
==================
USER INFORMATION
==================
lasttagtime : 0:20
transport : METRO
inout:IN
balance: 7100
terminal:c_1
탑승(0) 하차(1) : 0
IN
건대(b), 강남(c), 신림(d), 합정(e), 동역사(f)d
신림
1250원
승차 고고~~
태그할 경우 t 입력:
00:24
00:25
00:26
00:27
00:28
t
==================
USER INFORMATION
==================
lasttagtime : 0:24
transport : METRO
inout:IN
balance: 5850
terminal:d_2
탑승(0) 하차(1) : 1
OUT
건대(b), 강남(c), 신림(d), 합정(e), 동역사(f)f
동역사
하차시 추가요금 : 200원
```

지하철test(버스->지하철 환승 후 태그x 하차 후 지하철 탑승)

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:2 METRO IN 8950 b_1
0:4 BUS OUT 8750 a_1
0:8 METRO IN 8750 a_1
0:13 METRO OUT 8150 a_1
0:17 METRO IN 7100 d_1
0:20 METRO IN 7100 c_1
0:24 METRO IN 5850 d_2
0:28 METRO OUT 5650 d_2
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:2 METRO IN 8950 b_1
0:4 BUS OUT 8750 a_1
0:8 METRO IN 8750 a_1
0:13 METRO OUT 8150 a_1
0:17 METRO IN 7100 d_1
0:20 METRO IN 7100 c_1
0:24 METRO IN 5850 d_2
0:28 METRO IN 5650 a_2
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:2 METRO IN 8950 b_1
0:4 BUS OUT 8750 a_1
0:8 METRO IN 8750 a_1
0:13 METRO OUT 8150 a_1
0:17 METRO IN 7100 d_1
0:20 METRO IN 7100 c_1
0:24 METRO IN 5850 d_2
0:28 METRO IN 5650 a_2
0:31 METRO IN 3900 c_1
0:33 METRO OUT 3900 c_1
```

```
태그할 경우 t 입력:
00:28
00:29
00:30
00:31
t
==================
USER INFORMATION
==================
lasttagtime : 0:28
transport : METRO
inout:IN
balance: 5650
terminal:a_2
탑승(0) 하차(1) : 0
IN
건대(b), 강남(c), 신림(d), 합정(e), 동역사(f)c
강남
1750원
승차 고고~~
태그할 경우 t 입력:
00:31
00:32
00:33
t
==================
USER INFORMATION
==================
lasttagtime : 0:31
transport : METRO
inout:IN
balance: 3900
terminal:c_1
탑승(0) 하차(1) : 1
OUT
건대(b), 강남(c), 신림(d), 합정(e), 동역사(f)b
건대
하차시 추가요금 : 0원
```

KU KONKUK UNIVERSITY

지하철test(지하철->버스 환승 후 태그x 하차 후 지하철 탑승)

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:2 METRO IN 8950 b_1
0:4 BUS OUT 8750 a_1
0:8 METRO IN 8750 a_1
0:13 METRO OUT 8150 a_1
0:17 METRO IN 7100 d_1
0:20 METRO IN 7100 c_1
0:24 METRO IN 5850 d_2
0:28 METRO IN 5650 a_2
0:31 METRO IN 3900 c_1
0:33 METRO OUT 3900 c_1
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:2 METRO IN 8950 b_1
0:4 BUS OUT 8750 a_1
0:8 METRO IN 8750 a_1
0:13 METRO OUT 8150 a_1
0:17 METRO IN 7100 d_1
0:20 METRO IN 7100 c_1
0:24 METRO IN 5850 d_2
0:28 METRO IN 5650 a_2
0:31 METRO IN 3900 c_1
0:33 BUS IN 3900 b_1
```

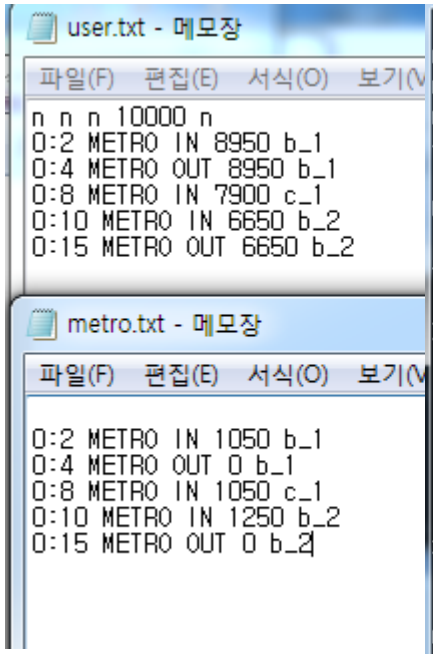
user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:2 METRO IN 8950 b_1
0:4 BUS OUT 8750 a_1
0:8 METRO IN 8750 a_1
0:13 METRO OUT 8150 a_1
0:17 METRO IN 7100 d_1
0:20 METRO IN 7100 c_1
0:24 METRO IN 5850 d_2
0:28 METRO IN 5650 a_2
0:31 METRO IN 3900 c_1
0:33 BUS IN 3900 b_1
0:39 METRO IN 2250 e_1
0:42 METRO OUT 2050 e_1
```

```
태그할 경우 t 입력:
00:33
00:34
00:35
00:36
00:37
00:38
00:39
t
==================
USER INFORMATION
==================
lasttagtime : 0:33
transport : BUS
inout:IN
balance: 3900
terminal:b_1
탑승(0) 하차(1) : 0
IN
건대(b), 강남(c), 신림(d), 합정(e), 동역사(f)e
합정
1650원
승차 고고~~
태그할 경우 t 입력:
00:39
00:40
00:41
00:42
t
==================
USER INFORMATION
==================
lasttagtime : 0:39
transport : METRO
inout:IN
balance: 2250
terminal:e_1
탑승(0) 하차(1) : 1
OUT
건대(b), 강남(c), 신림(d), 합정(e), 동역사(f)b
건대
하차시 추가요금 : 200원
```

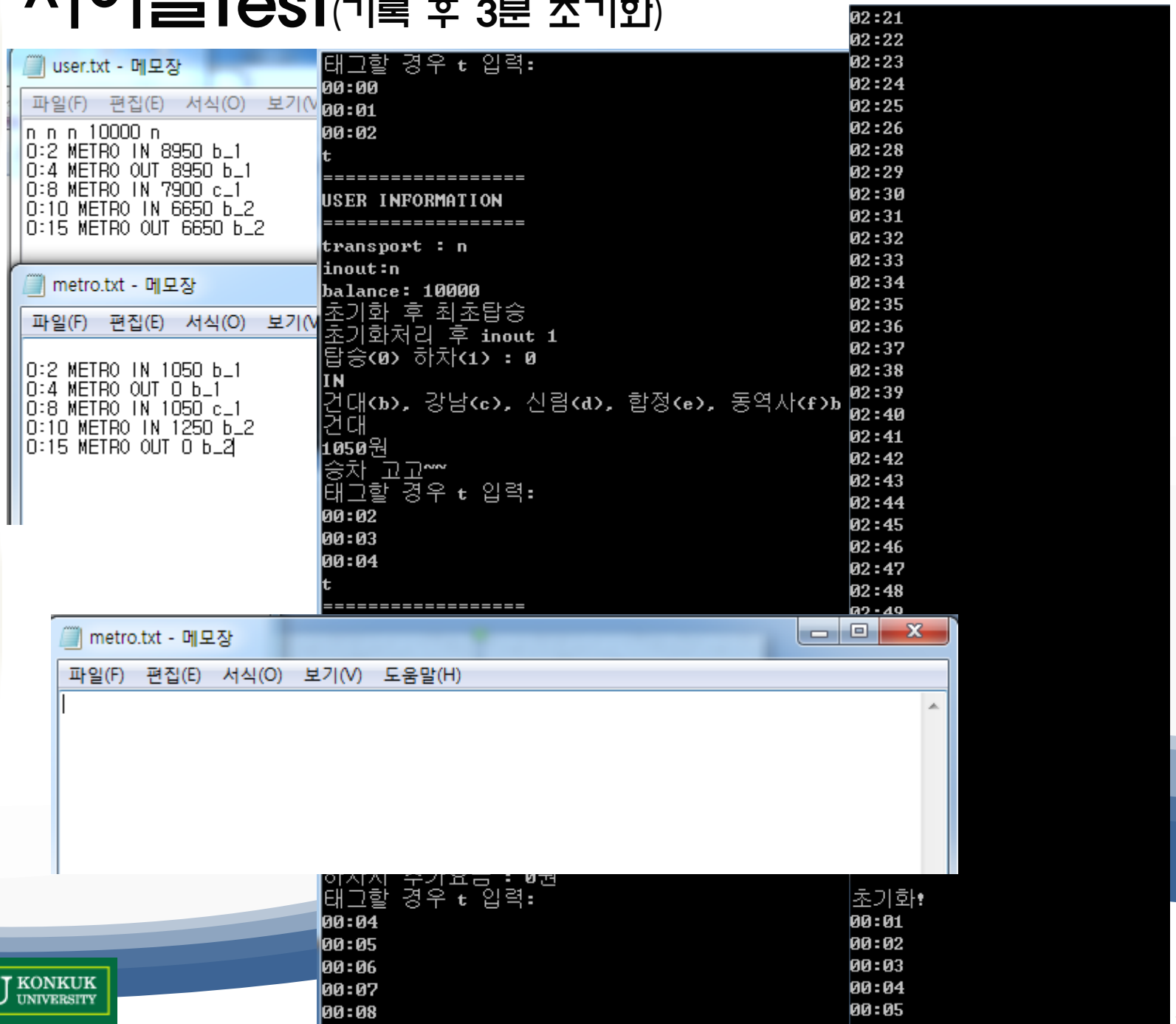
지하철test(기록 후 3분 초기화)




```
태그할 경우 t 입력:
00:00
00:01
00:02
t
==================
USER INFORMATION
==================
transport : n
inout:n
balance: 10000
초기화 후 최초탑승
초기화처리 후 inout 1
탑승(0) 하차(1) : 0
IN
건대(b), 강남(c), 신림(d), 합정(e), 동역사(f)b
건대
1050원
승차 고고~~
태그할 경우 t 입력:
00:02
00:03
00:04
t
==================
USER INFORMATION
==================
lasttagtime : 0:2
transport : METRO
inout:IN
balance: 8950
terminal:b_1
탑승(0) 하차(1) : 1
OUT
건대(b), 강남(c), 신림(d), 합정(e), 동역사(f)c
강남
하차시 추가요금 : 0원
태그할 경우 t 입력:
00:04
00:05
00:06
00:07
00:08
```

```
02:21
02:22
02:23
02:24
02:25
02:26
02:28
02:29
02:30
02:31
02:32
02:33
02:34
02:35
02:36
02:37
02:38
02:39
02:40
02:41
02:42
02:43
02:44
02:45
02:46
02:47
02:48
02:49
02:50
02:51
02:52
02:53
02:54
02:55
02:56
02:57
02:58
02:59
03:00

초기화!
00:01
00:02
00:03
00:04
00:05
```

지하철test(기록 후 3분 초기화)



버스test(초기화후 첫탑승)

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 BUS OUT 8950 a_1
```

C:₩Windows₩system32₩cmd.exe

```
태그할 경우 t 입력:00:00
00:01
00:02
00:03
t

TAG TIME 00:03
inout:n
transport : n
처음 입력시 IO: 1
balance: 10000
초기화 후 최초탑승
초기화처리 후 inout 1
탑승(0) 하차(1) : 0
IN


시간 : 0:3
금액 : 1050
잔액 : 8950


태그할 경우 t 입력:00:03
00:04
00:05
00:06
t

TAG TIME 00:06
transport : BUS
inout:IN
balance: 8950
terminal:a_1
탑승(0) 하차(1) : 1
OUT


시간 : 0:6
금액 : 0
잔액 : 8950
```

KU KONKUK UNIVERSITY

버스test(지하철하차 후 버스환승)

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 BUS OUT 8950 a_1
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 METRO OUT 8950 b_1
```

C:₩Windows₩system32₩cmd.exe

```
태그할 경우 t 입력:00:06
00:07
00:08
00:09
00:10
00:11
t

TAG TIME 00:11
transport : METRO
inout:OUT
balance: 8950
terminal:b_1
탑승(0) 하차(1) : 0
IN


시간 : 0:11
금액 : 0
잔액 : 8950


태그할 경우 t 입력:00:11
00:13
00:14
00:15
00:16
00:17
00:18
00:19
00:20
00:21
00:22
00:23
00:24
00:25
00:26
00:27
00:28
00:29
00:30
00:31
00:32
00:33
00:34
00:35
00:36
00:37
00:38
```

```
00:38
00:39
00:40
00:41
00:42
00:43
t

TAG TIME 00:43
transport : BUS
inout:IN
balance: 8950
terminal:b_1
탑승(0) 하차(1) : 1
OUT


시간 : 0:43
금액 : 100
잔액 : 8850
```

KU KONKUK UNIVERSITY

버스test(지하철하차 후 버스환승)

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 BUS OUT 8950 a_1
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 METRO OUT 8950 b_1
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 METRO OUT 8950 b_1
0:11 BUS IN 8950 b_1
0:43 BUS OUT 8850 b_1
```

C:₩Windows₩system32₩cmd.exe

```
태그할 경우 t 입력:00:06
00:07
00:08
00:09
00:10
00:11
t

TAG TIME 00:11
transport : METRO
inout:OUT
balance: 8950
terminal:b_1
탑승(0) 하차(1) : 0
IN


시간 : 0:11
금액 : 0
잔액 : 8950


태그할 경우 t 입력:00:11
00:13
00:14
00:15
00:16
00:17
00:18
00:19
00:20
00:21
00:22
00:23
00:24
00:25
00:26
00:27
00:28
00:29
00:30
00:31
00:32
00:33
00:34
00:35
00:36
00:37
00:38
```

하차시
30초 초과당 요금 100원

```
00:38
00:39
00:40
00:41
00:42
00:43
t

TAG TIME 00:43
transport : BUS
inout:IN
balance: 8950
terminal:b_1
탑승(0) 하차(1) : 1
OUT


시간 : 0:43
금액 : 100
잔액 : 8850
```

KU KONKUK UNIVERSITY

버스test(버스->지하철 환승 후 태그x 하차 후 버스 탑승)

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 METRO OUT 8950 b_1
0:11 BUS IN 8950 b_1
0:43 BUS OUT 8850 b_1
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 METRO OUT 8950 b_1
0:11 BUS IN 8950 b_1
0:43 METRO IN 8850 a_1
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 METRO OUT 8950 b_1
0:11 BUS IN 8950 b_1
0:43 METRO IN 8850 a_1
0:47 BUS IN 7200 a_2
0:51 BUS OUT 7200 a_2
```

```
태그할 경우 t 입력:00:43
00:44
00:45
00:46
00:47
t

TAG TIME 00:47
transport : METRO
inout:IN
balance: 8850
terminal:a_1
탑승(0) 하차(1) : 0
IN


시간 : 0:47
금액 : 1650
잔액 : 7200


태그할 경우 t 입력:00:47
00:48
00:49
00:50
00:51
t

TAG TIME 00:51
transport : BUS
inout:IN
balance: 7200
terminal:a_2
탑승(0) 하차(1) : 1
OUT


시간 : 0:51
금액 : 0
잔액 : 7200
```

KU KONKUK UNIVERSITY

버스test(지하철->버스 환승 후 태그x 하차 후 버스 탑승)

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 METRO OUT 8950 b_1
0:11 BUS IN 8950 b_1
0:43 BUS OUT 8850 b_1
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 METRO OUT 8950 b_1
0:11 BUS IN 8950 b_1
0:43 METRO IN 8850 a_1
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 METRO OUT 8950 b_1
0:11 BUS IN 8950 b_1
0:43 METRO IN 8850 a_1
0:47 BUS IN 7200 a_2
0:51 BUS OUT 7200 a_2
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 METRO OUT 8950 b_1
0:11 BUS IN 8950 b_1
0:43 METRO IN 8850 a_1
0:47 BUS IN 7200 a_2
0:51 BUS IN 7200 c_2
0:56 BUS IN 5450 a_3
0:59 BUS OUT 5450 a_3
```

C:₩Windows₩system32₩cmd.exe

```
태그할 경우 t 입력:00:51
00:52
00:53
00:54
00:55
00:56
t

TAG TIME 00:56
transport : BUS
inout:IN
balance: 7200
terminal:c_2
탑승(0) 하차(1) : 0
IN


시간 : 0:56
금액 : 1750
잔액 : 5450


태그할 경우 t 입력:00:56
00:57
00:58
00:59
t

TAG TIME 00:59
transport : BUS
inout:IN
balance: 5450
terminal:a_3
탑승(0) 하차(1) : 1
OUT


시간 : 0:59
금액 : 0
잔액 : 5450
```

버스test(버스 태그하차 후 버스 탑승)

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 METRO OUT 8950 b_1
0:11 BUS IN 8950 b_1
0:43 METRO IN 8850 a_1
0:47 BUS IN 7200 a_2
0:51 BUS IN 7200 c_2
0:56 BUS IN 5450 a_3
0:59 BUS OUT 5450 a_3
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 METRO OUT 8950 b_1
0:11 BUS IN 8950 b_1
0:43 METRO IN 8850 a_1
0:47 BUS IN 7200 a_2
0:51 BUS IN 7200 c_2
0:56 BUS IN 5450 a_3
0:59 BUS OUT 5450 a_3
1:3 BUS IN 4400 a_4
1:7 BUS OUT 4400 a_4
```

C:₩Windows₩system32₩cmd.exe

```
태그할 경우 t 입력:00:59
01:00
01:01
01:02
01:03
t

TAG TIME 01:03
transport : BUS
inout:OUT
balance: 5450
terminal:a_3
탑승(0) 하차(1) : 0
IN


시간 : 1:3
금액 : 1050
잔액 : 4400


태그할 경우 t 입력:01:03
01:04
01:05
01:06
01:07
t

TAG TIME 01:07
transport : BUS
inout:IN
balance: 4400
terminal:a_4
탑승(0) 하차(1) : 1
OUT


시간 : 1:7
금액 : 0
잔액 : 4400
```

KU KONKUK UNIVERSITY

버스test(버스 태그x 하차 후 버스 탑승)

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 METRO OUT 8950 b_1
0:11 BUS IN 8950 b_1
0:43 METRO IN 8850 a_1
0:47 BUS IN 7200 a_2
0:51 BUS IN 7200 c_2
0:56 BUS IN 5450 a_3
0:59 BUS OUT 5450 a_3
1:3 BUS IN 4400 a_4
1:7 BUS OUT 4400 a_4
```

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n
0:3 BUS IN 8950 a_1
0:6 METRO OUT 8950 b_1
0:11 BUS IN 8950 b_1
0:43 METRO IN 8850 a_1
0:47 BUS IN 7200 a_2
0:51 BUS IN 7200 c_2
0:56 BUS IN 5450 a_3
0:59 BUS OUT 5450 a_3
1:3 BUS IN 4400 a_4
1:7 BUS IN 4400 a_4
1:17 BUS IN 3350 a_5
1:21 BUS OUT 3350 a_5
```

```
태그할 경우 t 입력:01:07
01:08
01:09
01:10
01:11
01:12
01:13
01:14
01:15
01:16
01:17
t

TAG TIME 01:17
transport : BUS
inout:IN
balance: 4400
terminal:a_4
탑승(0) 하차(1) : 0
IN


시간 : 1:17
금액 : 1050
잔액 : 3350


태그할 경우 t 입력:01:17
01:18
01:19
01:20
01:21
t

TAG TIME 01:21
transport : BUS
inout:IN
balance: 3350
terminal:a_5
탑승(0) 하차(1) : 1
OUT


시간 : 1:21
금액 : 0
잔액 : 3350
```

KU KONKUK UNIVERSITY

버스test(기록 후 3분 초기화)

user.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
n n n 10000 n

0:2 BUS IN 8950 a_1
0:3 BUS OUT 8950 a_1
0:7 BUS IN 7900 a_2
0:10 BUS OUT 7900 a_2
0:14 BUS IN 6850 a_3
0:16 BUS OUT 6850 a_3
```

bus.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
0:2 BUS IN 1050 a_1
0:3 BUS OUT 0 a_1
0:7 BUS IN 1050 a_2
0:10 BUS OUT 0 a_2
0:14 BUS IN 1050 a_3
0:16 BUS OUT 0 a_3
```

C:₩Windows₩system32₩cmd.exe

```
태그할 경우 t 입력:00:00
00:01
00:02
t

TAG TIME 00:02
inout:n
transport : n
처음 입력시 IO: 1
balance: 10000
초기화 후 최초탑승
초기화처리 후 inout 1
탑승(0) 하차(1) : 0
IN


시간 : 0:2
금액 : 1050
잔액 : 8950


태그할 경우 t 입력:00:02
00:03
t

TAG TIME 00:03
transport : BUS
inout:IN
balance: 8950
terminal:a_1
탑승(0) 하차(1) : 1
OUT


시간 : 0:3
금액 : 0
잔액 : 8950
```

C:₩Windows₩system32₩cmd.exe

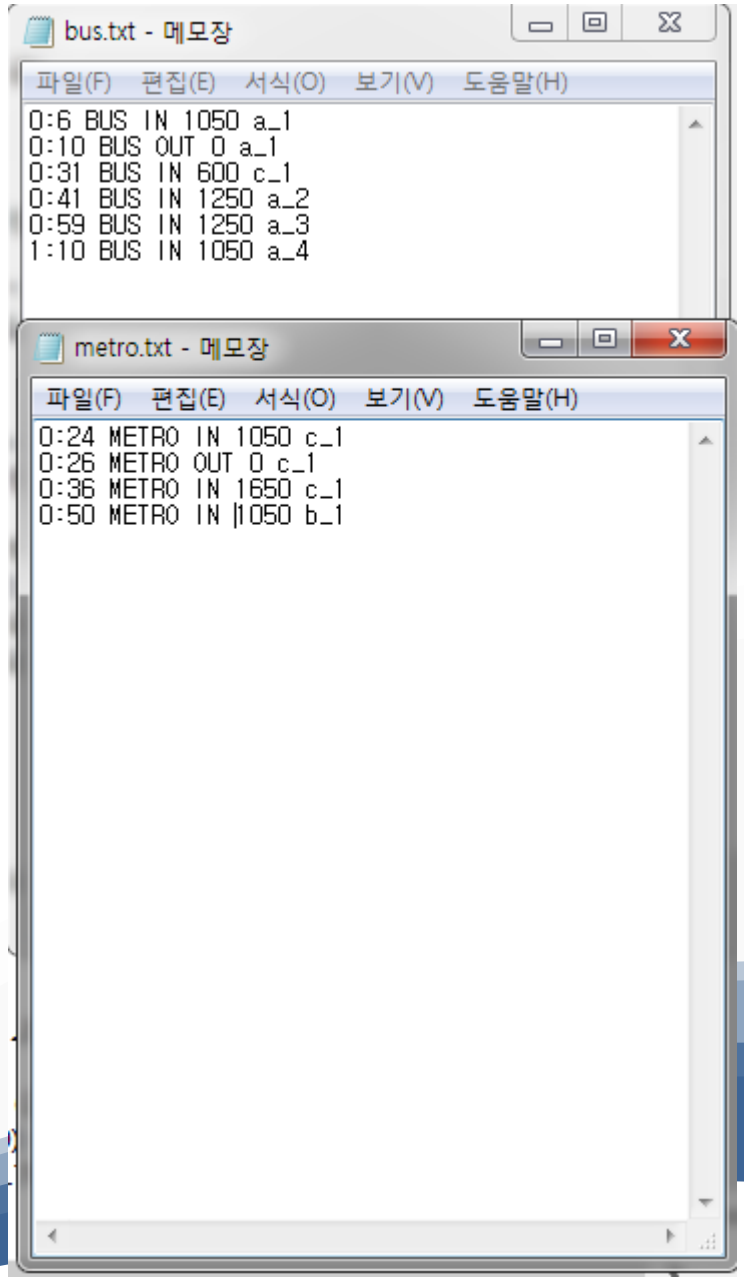
```
02:40
02:41
02:42
02:43
02:44
02:45
02:46
02:47
02:48
02:49
02:50
02:51
02:52
02:53
02:54
0
02:54
02:55
02:56
02:57
0
02:57
02:58
02:59
03:00

초기화!
00:01
00:02
00:03
00:04
```

bus.txt - 메모장

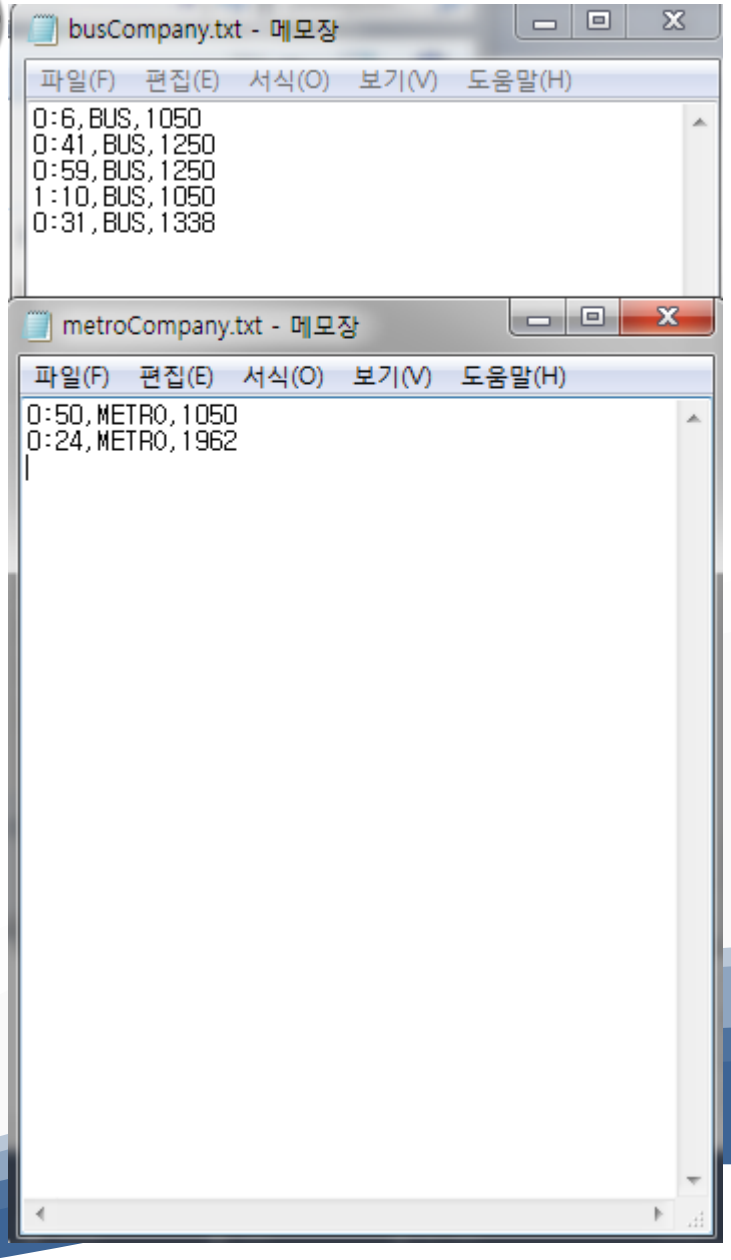
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

KU KONKUK UNIVERSITY

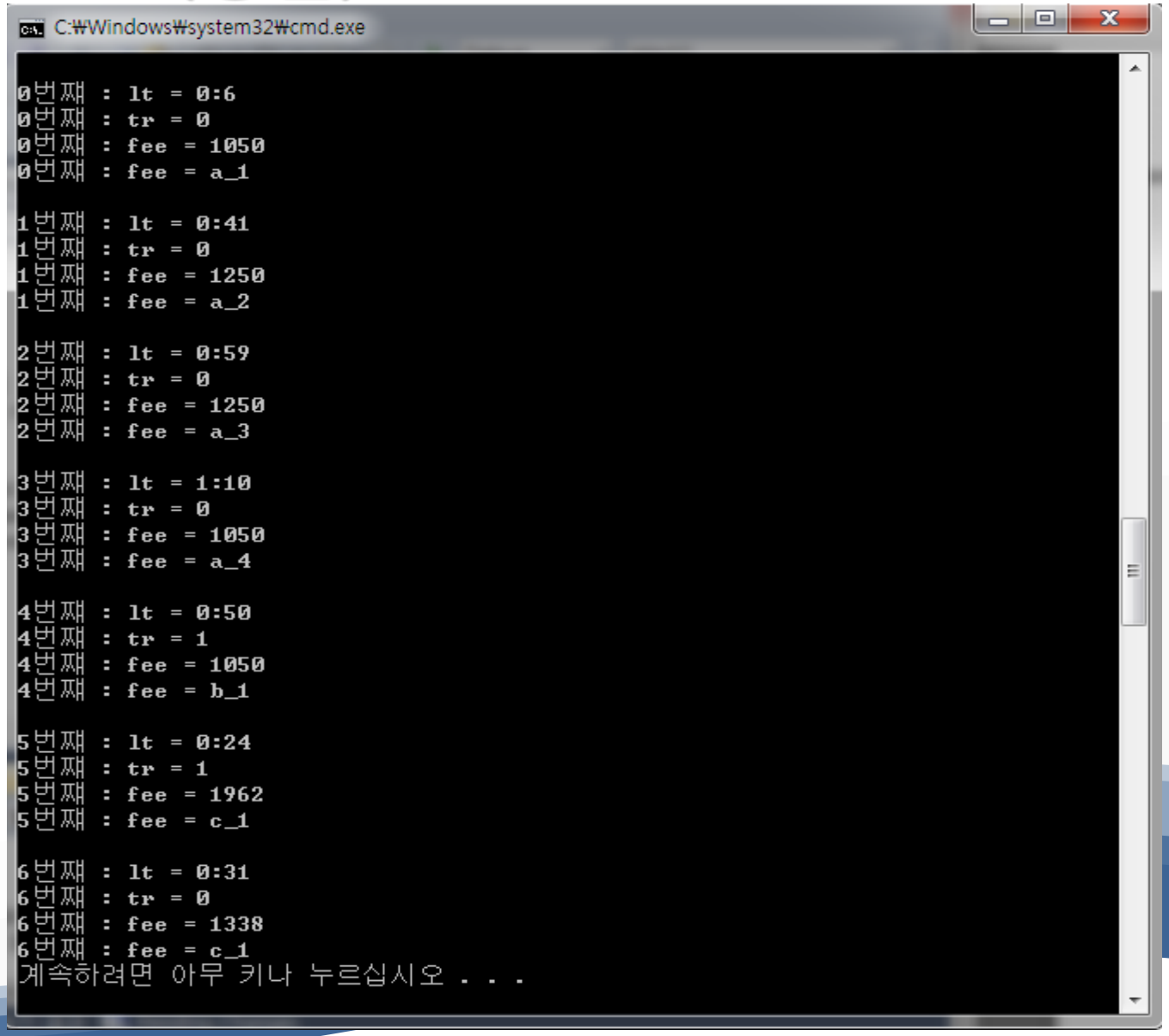
Unit Test(정산)


bus.txt - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
0:6 BUS IN 1050 a_1
0:10 BUS OUT 0 a_1
0:31 BUS IN 600 c_1
0:41 BUS IN 1250 a_2
0:59 BUS IN 1250 a_3
1:10 BUS IN 1050 a_4
metro.txt - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
0:24 METRO IN 1050 c_1
0:26 METRO OUT 0 c_1
0:36 METRO IN 1650 c_1
0:50 METRO IN 1050 b_1

Unit Test(정산)


busCompany.txt - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
0:6,BUS,1050
0:41,BUS,1250
0:59,BUS,1250
1:10,BUS,1050
0:31,BUS,1338
metroCompany.txt - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
0:50,METRO,1050
0:24,METRO,1962

Unit Test(정산)


C:₩Windows₩system32₩cmd.exe
0번째 : lt = 0:6
0번째 : tr = 0
0번째 : fee = 1050
0번째 : fee = a_1
1번째 : lt = 0:41
1번째 : tr = 0
1번째 : fee = 1250
1번째 : fee = a_2
2번째 : lt = 0:59
2번째 : tr = 0
2번째 : fee = 1250
2번째 : fee = a_3
3번째 : lt = 1:10
3번째 : tr = 0
3번째 : fee = 1050
3번째 : fee = a_4
4번째 : lt = 0:50
4번째 : tr = 1
4번째 : fee = 1050
4번째 : fee = b_1
5번째 : lt = 0:24
5번째 : tr = 1
5번째 : fee = 1962
5번째 : fee = c_1
6번째 : lt = 0:31
6번째 : tr = 0
6번째 : fee = 1338
6번째 : fee = c_1
계속하려면 아무 키나 누르십시오 . . .

Q & A

KU KONKUK UNIVERSITY