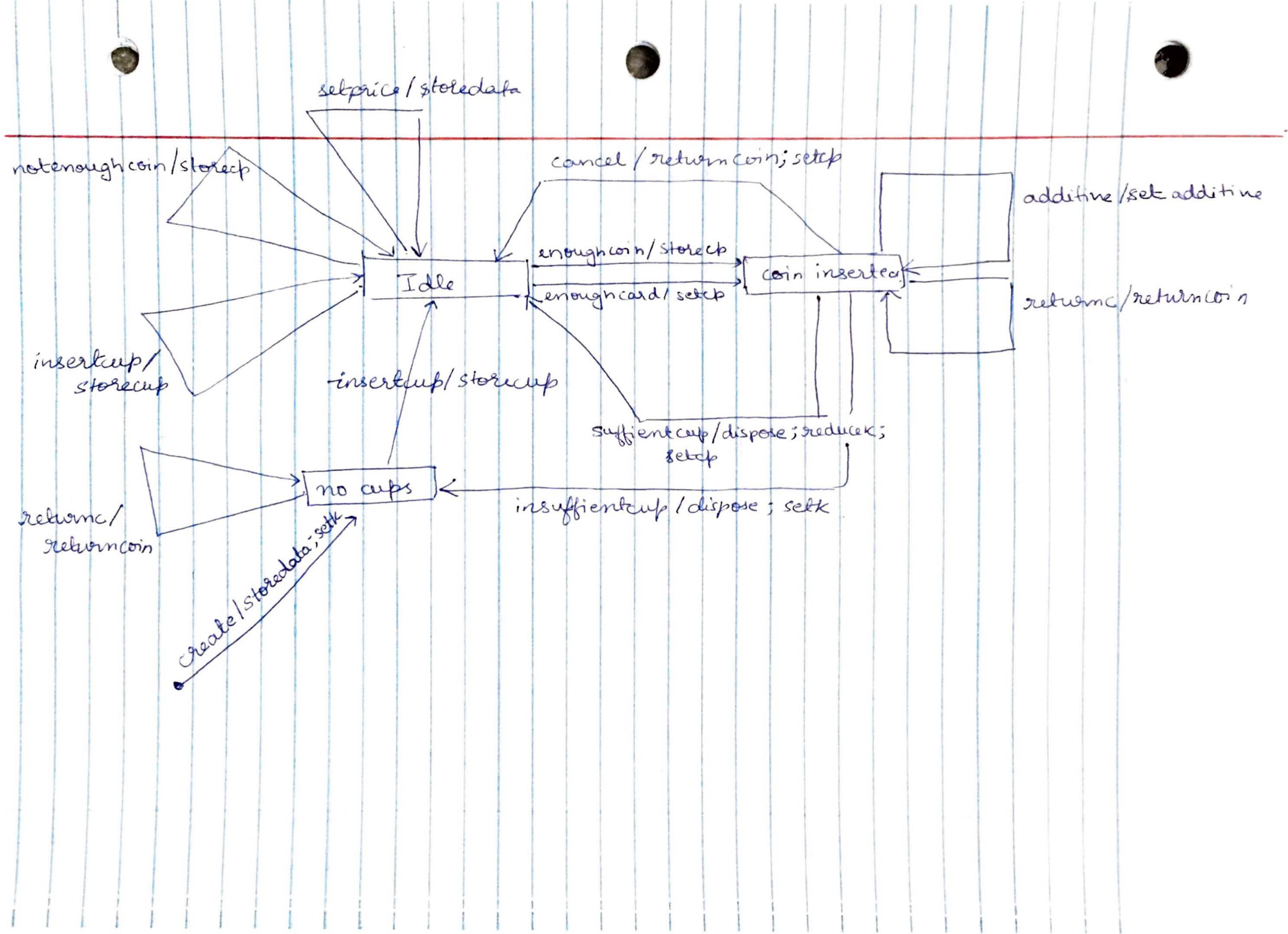
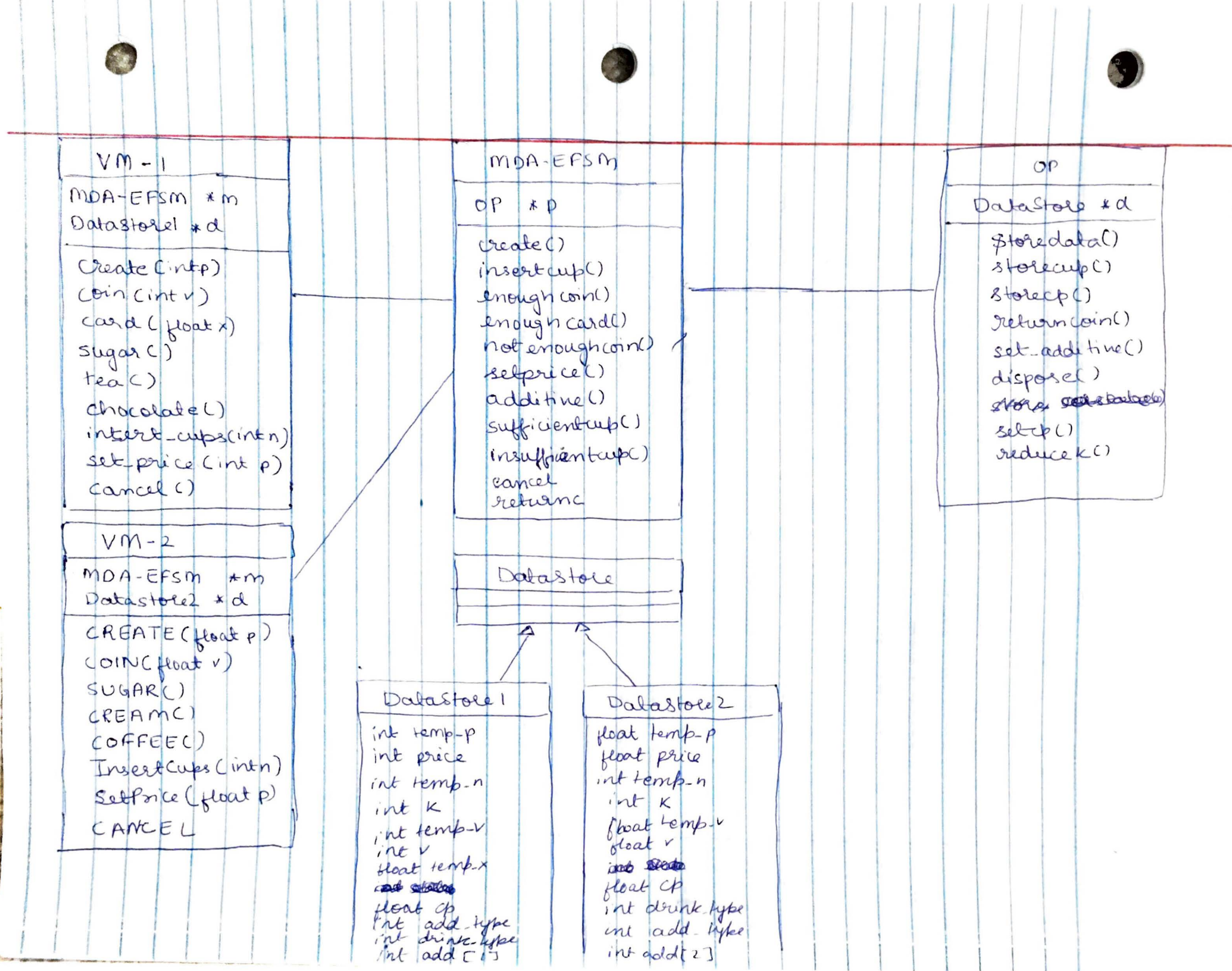

setprice / storedata
notenoughcoin / storecp
cancel / returncoin; setcp
additive / set additive
Idle
enoughcoin / storecp
enoughcard / setcp
coin inserted
returnc / returncoin
insertcup / storecup
insertcup / storecup
sufficientcup / dispose; reducek; setcp
no cups
insufficientcup / dispose; setk
returnc / returncoin
create / storedata; setk


VM - 1
MDA-EFSM *m
DataStore1 *d
Create(int p)
Coin(int v)
card(float x)
sugar()
tea()
chocolate()
insert_cups(int n)
set_price(int p)
cancel()
VM - 2
MDA-EFSM *m
Datastore2 *d
CREATE(float p)
COINC(float v)
SUGAR()
CREAMC()
COFFEEC()
InsertCups(int n)
SetPrice(float p)
CANCEL
MDA-EFSM
OP *p
create()
insertcup()
enoughcoin()
enoughcard()
notenoughcoin()
selprice()
additive()
sufficientcup()
insufficientcup()
cancel
returnc
OP
DataStore *d
Storedata()
storecup()
storecp()
returncoin()
set_additive()
dispose()
setcp()
reducek()
Datastore
Datastore1
int temp_p
int price
int temp_n
int k
int temp_v
int v
float temp_x
float cp
int add_type
int drink_type
int add[1]
Datastore2
float temp_p
float price
int temp_n
int k
float temp_v
float v
float cp
int drink_type
int add_type
int add[2]

Input - Processor VM-1

```
create ( int p)
    d → temp.p = p
    m → create()

insert_cups (int n)
    if n > 0 then
        d → temp_n = n
        m → insertcup()

coin (int v)
    d → temp_v = v
    if d → k > 0 then
        if v + d → cp ≥ d → price()
            m → enoughcoin()
        else
            m → notenoughcoin()
    else
        m → returnc()

card ( float x)
    d → temp_x = x
    if x ≥ d → price
        m → enoughcard()

set_price (int p)
    m → setprice()
```

Input - Processor VM-2

```
CREATE ( float p)
    d → temp.p = p
    m → create()

InsertCups ( int n)
    if n > 0 then
        d → temp_n = n
        m → insertcup()

COIN ( float v)
    d → temp_v = v
    if d → k > 0 then
        if v + d → cp ≥ d → price()
            m → enoughcoin()
        else
            m → notenoughcoin()
    else
        m → returnc()

SetPrice (float p)
    m → setprice()
```

```
tea()
    if d->k>1
    d->drink_type = 0
    if d->k > 1
        m->suffientcup()
    else
        m->insuffientcup()

chocolate()
    d->drink_type = 1
    if d->k > 1
        m->suffientcup()
    else
        m->insuffientcup()

cancle()
    m->cancel

tea
sugar()
    d->add_type = 0
    m->additive()
```

```
(coffee)
COFFEE()
    d->drink_type = 2
    if d->k > 1
        m->suffientcup()
    else
        m->insuffientcup()

CANCEL()
    m->cancel

SUGAR()
    d->add_type = 0
    m->additive()

CREAM()
    d->add_type = 1
    m->additive()
```

meta-event

create()
returnC()
insertcup()
notenoughcoin()
enoughcoin()
enoughcard()
selprice()
cancel()
suffientcups()
insuffientcups()
additive()

meta action

storedata // stores price for item from temporary data variable
~~xxxxxx~~ // ~~stores state variable to 1 if state==0 else 0 if state!=1~~
returncoin // returns the cash
storecup // updates the value of k
storecp // updates the value of cp
setcp // Initilizes cp to 0
set K // Initilize K to 0
reducek // Decrements the value of k by 1
Setadditive // Sets the specified index to 0 if 1 or otherwise in the add array here 0 index ports to sugar 1 to cream
dispose // → Add additives based on add[] and dispose drink of drink_type