Fall 2015.

Point Of Sale Application

OOP244 Assignment V4.1

(V4.0 Milestone 4)

(V4.1 Changed Perishable signature to P)

-
-

CLASSES TO BE DEVELOPED

Date

PosIO

Item

Perishable

NonPerishable

PosSys

PROJECT DEVELOPMENT PROCESS

•
•
•
•
•

FILE STRUCTURE FOR THE PROJECT

```
TAX (0.13)              The tax rate for the goods
MAX_SKU_LEN (7)         The maximum size of an SKU code

MIN_YEAR (2000)         The min year used to validate year input
MAX_YEAR (2030)         The max year used to validate year input

MAX_NO_ITEMS (2000)      The maximum number of records in the data file.
```

```
#ifndef SICT_HeaderFileName_H__
#define SICT_HeaderFileName_H__


#endif
```

```
#ifndef SICT_POSSYS_H__
#define SICT_POSSYS_H__
```

MILESTONE 1: THE DATE CLASS

`_dateOnly` `bool _dateOnly;`

Member Data (attributes):

```
int _year;

int _mon;
int _day;

int _hour;
int _min;
int _readErrorCode;
```

```
          NO_ERROR    0  -- No error - the date is valid
          CIN_FAILED  1  -- istream failed on accepting information
          YEAR_ERROR  2  -- Year value is invalid
          MON_ERROR   3  -- Month value is invalid
          DAY_ERROR   4  -- Day value is invalid
          HOUR_ERROR  5  -- Hour value is invalid
          MIN_ERROR   6  -- Minute value is invalid
```

`bool _dateOnly;` A flag that is true if the object is to only hold the date and not the time.

Private Member functions (private methods):

```
int value()const; (this function is already implemented and provided)
```

```
void errCode(int errorCode);

void set(int year, int mon, int day, int hour, int min);
```

NO_ERROR.

Constructors:

`void set();`

Public member-functions (methods) and operators:

```
      bool operator==(const Date& D)const;
      bool operator!=(const Date& D)const;
      bool operator<(const Date& D)const;
      bool operator>(const Date& D)const;
      bool operator<=(const Date& D)const;
      bool operator>=(const Date& D)const;
```

`operator<`

`this->value()` `D.value()`

`int mdays()const;` (this function is already implemented and provided)

`void set();` (this function is already implemented and provided)

Accessor or getter member functions (methods):

```
int errCode()const;
bool bad()const;
bool dateOnly()const;
void dateOnly(bool value);
```

IO member-funtions (methods):

```
std::istream& read(std::istream& is = std::cin);
```

`(istr)`

`_readErrorCode` `CIN_FAILED`

```
                                              (istr)
```

```
_readErrorCode
```

```
(istr)
```

```
std::ostream& write(std::ostream& ostr = std::cout)const;
```

```
                                            (ostr)
```

```
                                      (istr)
```

Non-member IO operator overloads: (Helpers)

Preliminary task

MILESTONE 2: THE POSIO INTERFACE V1.0

fstream iostream

Pure virtual member functions (methods):

In future milestones children of PosIO will implement this method, when they are to be stored in a file.

In future milestones children of PosIO will implement this method, when they are to be read from a file.

In future milestones children of PosIO will implement this method when they are to be printed on the screen in two different formats:
***Linear:** the class information is to be printed in one line*
***Form:** the class information is to be printed in several lines like a form.*

In future milestones children of PosIO will implement this method when their information is to be received from console.

Submission:

MILESTONE 3: THE ITEM CLASS

_sku:

_name:

_price

_taxed:

_quantity:

Public member functions and constructors

-
-
-
-
-

Copy Constructor;

See below:

Dynamic memory allocation necessities

Accessors

Setters:

- **sku**
- **price**
- **name**
- **taxed**
- **quantity**

Getters:

- **sku**, returns constant character pointer
- **price**, returns double
- **name**, returns constant character pointer
- **taxed**, returns boolean
- **quantity**, returns integer

- **cost**, returns double

- **isEmpty** returns bool

Member Operator overloads:

Operator== : receives a constant character pointer and returns a Boolean.

Operator+= : receives an integer and returns an integer.

Operator-= : receives an integer and returns an integer.

Non-Member operator overload:

Operator+= :

Non-member IO operator overloads:

Submission:

MILESTONE 4: THE NONPERISHABLE AND PERISHABLE CLASSES

Part one: NonPerishable

ErrorMessage

ErrorMessage();

void clear();

bool isClear()const;

void message(const char* value);

const char* message()const;

NonPerishable Class

Private member variables

_err

Constructor:

Public member functions

std::fstream& save(std::fstream& file)const:

“N”

file

```
          sku, name, price, taxed, quantity
```

```
N,1234,Candle,1.23,1,38<Newline>
```

std::fstream& load(std::fstream& file)

N,"

Hint: create temporary variables of type double, int and string and read the fields one by one, skipping the commas. After each read, set the member variables using setter methods.

std::ostream& write(std::ostream& ostr, bool linear)const

_err

_err

Linear is true:

```
1234   |Candle              |   1.23| t |  38|    52.82|
```

SKU:
Name:
Price:
Taxed:
Quantity:
Cost:
One Bar NO NEW LINE

Linear is false:

```
Name:
Candle
Sku: 1234
Price: 1.23
Price after tax: 1.39
```

```
Quantity: 38
Total Cost: 52.82 <Newline>

OR if not taxed

Sku: 1234
Name: Candle
Price: 1.23
Price after tax: N/A
Quantity: 38
Total Cost: 46.74 <Newline>
```

std::istream& read(std::istream& istr):

```
Non-Perishable Item Entry:
Sku: 1234<ENTER>
Name:
Candle<ENTER>
Price: 1.23<ENTER>
Taxed: y<Enter>
Quantity: 38<ENTER>
```

istr fail

_err

Invalid Price Entry
Invalid Taxed Entry, (y)es or (n)o
Invalid Quantity Entry

```
istr.setstate(ios::failbit);
```

Part Two: Perishable Class

Perishable Class

Please note that the Perishable and NonPerishable classes are identical in logic. The only difference is that the Perishable class has one extra member variables that have to be received and printed (in addition to the variables in an Item).

Private member variables

- _err
-

Constructor:

Public member functions

Public Accessors (setters and getters)

const Date& expiry()const;

void expiry(const Date &value);

Pure virtual method implementations

std::fstream& save(std::fstream& file)const:

P”

file

```
sku, name, price, taxed, quantity, expiry date
```

```
P,1234,4L Milk,3.99,0,2,2015/12/10<Newline>
```

std::fstream& load(std::fstream& file)

load P,

Hint: create temporary variables of type double, int, string and Date, then read the fields one by one, skipping the commas. After each read set the member variables using setter methods.

std::ostream& write(std::ostream& ostr, bool linear)const:

_err

_err

Linear is true:

```
1234   |4L Milk            |   3.99|  p|   2|     7.98|
```

Sku:
Name:
Price:
Taxed:
Quantity:
Cost:
NO NEW LINE

Linear is false:

```
Name:
4L Milk
Sku: 1234
Price: 3.99
Price after tax: 4.51
Quantity: 2
Total Cost: 9.02
Expiry date: 2015/12/10 <Newline>
```

```
Name:
4L Milk
```

```
Sku: 1234
Price: 3.99
Price after tax: N/A
Quantity: 2
Total Cost: 7.98
Expiry date: 2015/12/10 <Newline>
```

std::istream& read(std::istream& istr):

```
Perishable Item Entry:
Sku: 1234<ENTER>
Name:
4L Milk<ENTER>
Price: 3.99<ENTER>
Taxed: n<ENTER>
Quantity: 2<ENTER>
Expiry date (YYYY/MM/DD) : 2015/12/10<ENTER>
```

istr fail

_err

Invalid Price Entry
Invalid Taxed Entry, (y)es or (n)o
Invalid Quantity Entry

```
istr.setstate(ios::failbit);
```

_err

CIN_FAILED:	**Invalid Date Entry**
YEAR_ERROR:	**Invalid Year in Date Entry**
MON_ERROR:	**Invalid Month in Date Entry**
DAY_ERROR:	**Invalid Day in Date Entry**

```
istr.setstate(ios::failbit);
```

Submission: