2023年度 自由課題 コンビニのレジシステムを実装

34714117
林慶宏

2023年12月21日

1 はじめに

今回の自由課題では、コンビニのレジシステムを実装する。こう考えたきっかけとしては、私はコンビニでアルバイトをしていて、そこで頻繁に触るレジのシステムについて、どうすれば実装できるのだろうかと疑問に思ったからである。

2 関数の説明

このセクションでは、プログラム内の各関数について詳細に説明。

2.1 Item 構造体

この構造体は、商品情報を管理するための連結リストの基本的な構造を定義している。ここで定義された Item という構造体は、商品の名前、カテゴリー、価格、そして購入された回数といった情報を保持するためのフィールドを含んでいる。さらに、この構造体には Item *next というフィールドもあり、これは次の Item 構造体へのポインタとして機能する。このような設計は、商品情報を連結リストとして効率的に管理するために用いられる。

Item *head = NULL; という行は、この連結リストの先頭を指し示すポインタ head を宣言し、初期値として NULL を設定している。これは、リストが空であることを意味している。このリストに新しい Item が追加されるたびに、head ポインタは新しく追加された Item を指すように更新される。

このコードがこのように設計された理由は、商品情報の管理を柔軟にし、リストのサイズを必要に応じて動的に変更できるようにするためである。各 Item が次の Item を指すことで、連続したデータの流れを作り出し、リスト全体を通じてデータを追跡しやすくなる。このような構造は、商品情報が頻繁に更新される状況や、商品の数が事前に未定である場合に特に有効である。

2.2 loadItemsFromFile 関数

loadItemsFromFile 関数は、指定されたファイルから商品情報を読み込み、それを連結リストの形でメモリに格納するためのものである。この関数はまず、指定されたファイルを読み込みモードで開く。フファイルが正常に開けた場合、関数はファイルの終わりまでデータの読み込みを続ける。この読み込みプロセスは feof 関数を使用してファイルの終わりを検出することで行われる。各繰り返し処理では、新しい Item 構造体を,malloc によって動的に確保し、fscanf を使用してファイルから商品名、カテゴリ、価格を読み込む。これらのデータが正しく読み込まれた場合（fscanf の戻り値が 3 である場合）、新しい Item の purchased フィールドを 0 に設定し、next ポインタを NULL にする。そして、この新しいアイテムを連結リストに追加する。リストが空の場合、新しいアイテムがリストの先頭になり、既にアイテムが存在する場合はリストの最後に追加される。fscanf からの戻り値が 3 以外の場合、ファイルの読み込みに問題があったと判断し、新しく確保したメモリを解放し、読み込みプロセスを中断する。関数の最後に、ファイルは fclose 関数を使用して閉じる。

2.3 構造体 money

この C 言語のコードは、レジ内の紙幣と硬貨の情報を管理するためのデータ構造を定義している。Money という名前の構造体を用いており、この構造体は各種類の紙幣と硬貨の枚数を保持するためのフィールドが含まれている。具体的には、10000 円札、5000 円札、2000 円札、1000 円札、500 円玉、100 円玉、50 円玉、10 円玉、5 円玉、1 円玉それぞれの枚数を整数型で保持している。このような構造体を定義することで、レジ内にある異なる額面の紙幣と硬貨の枚数を一つの変数内で集約的に管理することが可能になる。これにより、レジの金銭管理が容易になり、必要に応じて各額面の枚数を簡単に追加、削除、確認することができる。コード内で定義されている inside_money という変数は、Money 構造体のインスタンスであり、初期値として各種額面の紙幣と硬貨の枚数が設定されている。この変数はレジ内の現在の紙幣と硬貨の枚数を表しており、プログラム内でレジの状態を追跡する際に使用される。

2.4 add,submoney 関数,money_sum 関数

money_sum 関数は、引数として与えられた Money 構造体内の各額面の紙幣と硬貨の枚数に基づき、その総額を計算して返す。この関数は、各額面の紙幣と硬貨の枚数にそれぞれの額面の金額を乗じ、その合計を計算する。この関数は、レジ内の現金の合計金額を簡単に計算するために役立つ。subMoney 関数は、2 つの Money 構造体のポインタを引数として受け取り、第一引数の Money 構造体から第二引数の Money 構造体の額を引く。addMoney 関数もまた、2 つの Money 構造体のポインタを引数として受け取り、第一引数の Money 構造体に第二引数の Money 構造体の額を加える。ここで、引数にポインタを採用した理由としては、この関数内で行われた、money 構造体の更新が、関数の外側にも反映されるようにするためである。その関数内では構造体のコピーが行われるため、関数内で行われた更新は、関数の外側には反映されない。

2.5 printmoney 関数

この関数は、構造体 money の中身を表示するための関数である。

2.6 calculateChange 関数

この関数はお釣りの計算を行う関数である。関数の中で、最初に Money 型のローカル変数 change が宣言され、その全ての要素が 0 で初期化されている。これはお釣りの各種類の紙幣と硬貨の数を保持するため。その後、複数の while ループがあり、それぞれのループは異なる額面の紙幣や硬貨に対してお釣りが可能かどうかを判断し、可能な場合はその額面の数を増やし、money からその額面を減算する。このプロセスは、money がその額面よりも小さくなるまで続けられる。2000 円紙幣の処理がコメントアウトしている。私の働いているコンビニのレジは、2000 円札の表記はあるが、2000 円札が一般的に使用されていないため、お釣りとして扱わないという意図がある。最後に、subMoney 関数を呼び出して inside_money から計算されたお釣り change を減算し、printMoney 関数を使ってお釣りの詳細を出力。これにより、使用された紙幣や硬貨の数をユーザーが確認できるようになっている。

2.7 register_items 関数

の関数は、ユーザーからの商品名の入力を受け、対応する商品が既存のリストに存在するかを確認し、存在する場合はその商品の価格情報を用いて合計価格と消費税の合計を更新する役割を持つ。関数の動作は、まずユーザーに商品名の入力を促し、その入力を受け取る。ここで、ユーザーが入力した商品名は item_name という文字配列に格納。次に、この商品名が連結リスト内のどの Item 型のノードと一致するかを確認するための線形検索が行われる。これを、strcmp で行っている。

もし入力された商品名に一致する商品がリスト内に見つからなかった場合、関数は商品が見つからなかったことをユーザーに通知する。一方で、商品が見つかった場合には、その商品が購入されたとして処理され、purchased フラグが 1 に設定される。また、商品のカテゴリに応じて、適切な消費税率（標準税率または軽減税率）が適用され、商品価格から消費税の額が計算。この計算結果は、合計価格と消費税の合計に反映される。

合計価格は、商品の価格から計算された消費税額を差し引いた値を加算することで更新される。このプロセスは、ユーザーが登録を終了することを選択するまで繰り返されます。

2.8 receive_payment 関数

この関数は、引数として、客から受け取ったお金を表す Money 構造体の、receive_money に、受け取ったお金を標準入力を用いて受け取る関数である。この関数の引数も、引数の変更が関数の外側にも反映されるように、ポインタを用いている。

2.9 printReceiptToFile 関数

この C 関数は、購入された商品のレシートをテキストファイルとして出力する機能を持っている。購入された商品の情報、合計金額、消費税額、受け取った金額、お釣りの情報を受け取り、それらをファイルに出力する。まず、この関数は現在の日時を取得する。これはライブラリ¡time_h¿を使用している。まず、時間を格納している構造体 time_t を取得し、time_t 型のオブジェクトを現在の時間帯で struct tm に変換する。time_t t と struct tm *local を使って、現在の日時を取得し、ローカルタイムに変換。これ、によってレシートに日付と時刻を記録する事ができた。

次に、ファイル名を生成。ファイル名は現在の年、月、日、時間、分を含む形式で、sprintf 関数を使用してフォーマットする。こうすることで、各レシートが独自のファイル名を持ち、識別しやすくなる。

その後、fopen 関数を使用してテキストファイルを書き込みモードで開きます

ファイルが正常に開かれた後、レシートの内容がファイルに書き込まれる。これには、日時、購入された商品のリスト（商品名と価格）、合計金額、消費税、受け取った金額、お釣りの情報が含まれる。商品リストは、purchasedItems ポインタを使ってリンクリストをループ処理し、各商品の情報を出力する。

最後に、ファイルを閉じる

2.10 calculate 関数

この関数は、今まで作成した関数を用いて、レジの処理を行う関数である。流れとしては、まず、一回の会計での合計金額と消費税額、受け取り硬貨紙幣を格納する変数を宣言する。まずは買う商品を登録する。そのために register_items を呼び出す。その次に、受け取ったお金を入力する。そのために receive_payment を呼び出す。受け取ったお金をレジ内金に入れる。受け取ったお金が購入商品学より大きいか判別する。その後、お釣りを計算する。そのために calculateChange を呼び出す。最後に、レシートを出力する。そのために printReceiptToFile を呼び出す。また、引数に total_price と tax_price のポインタ型を持つ。これは、1 日の合計売上と消費税を格納する変数だと意図している。そして最後に、それらの変数に」合計金額と消費税額を加算する。

2.11 check 関数

この関数は、レジ業務の「在高点検」という業務を再現するために作ったものである。「在高点検」とは、通常のレジ業務以外で不正なお金の出し入れがないかを確認するものである。この関数は、レジ内の紙幣と硬貨の合計金額を計算し、最初に元々あったお金と、売上金額を比較することで、不正なお金の出し入れがないかを確認する。こちらは値を確認するものであり、値を変更するものではないため、引数にポインタを用いていない。

2.12 settlement 関数

この関数は、レジ業務の「精算」という業務を再現するために作ったものである。「精算」とは、、ある時間内の間に売り上げた金額を数えること。通常は 1 日に一回行う。これを行うと、売上金額と消費税が 0 にリセットされる。これについては、値を更新するものであるため、引数にポインタを用いている。

2.13 refund 関数

この関数は、特定の商業取引における返金処理を行うもの。引数には今日の総売上、消費税額、返金額、およびレジ内の現金状況を示す変数が含まれている。

まず、顧客から返品された商品名を入力させる。次に、この商品名が商品リストに存在するかを確認するために、リストを順に検索。もし商品がリストになければ、商品が見つからなかった旨のメッセージを表示し。見つかった場合には、返品される商品の名前と価格を表示し、返金額を計算。

返金処理では、まずレジ内の現金状況を更新。次に、今日の総売上から返金額を減算。これは、返品によって総売上が実質的に減少するため。

さらに、消費税の計算も行われます商品のカテゴリに基づいて適切な消費税率を適用し、今日の総消費税から返金に相当する税額を減算。最後に、返金総額に今回の返金額を加算。

2.14 inside_money_check 関数

この関数は、レジ内のお金の枚数を確認し、レジ内のお金の枚数が少ない場合は、コンビニ事務所の金庫から補充する業務を再現したものである。基本的に、1 万円を事務所の金庫から持ってきて、それを両替して補充する。

2.15 main 関数

最初に商品情報の入ったファイルを読み込む。そのために loadItemsFromFile 関数を呼び出す。ファイル名は”items.txt” これは、今まで作成した関数を実行するプラットホームの役割をもつ。最初にメニュー画面を表示し、その中からモードを選択する。1-5 は、基本的に今まで作成した関数を呼び出すのみ。6 は、繰り返しループから脱してプログラムを終了する。各モード終了時に、inside_money_check 関数を呼び出し、レジ内のお金の枚数を確認する。また、変数として、一日の売上、消費税、返金額、収納代行（公共料金の支払いを代行するサービス）件数などの変数を持っている。収納代行については、今回は実装しなかった。

3 感想

今回の演習は、とても骨太なもので、非常に大変だった。自分のバイト先のレジシステムがいかに複雑で多機能だったかを実感した。それでも、自分が当初実装したかった機能を諦めたことも多かった。収納代行サービスの実装を諦め、カード決済やバーコード決済など、現代のコンビニで当たり前のように使われている機能を実装することができなかった。あと、商品の売上ランキングなども実装したかったが、私の「データ構造とアルゴリズム」関連の知識、演習不足なため、連結リストでのソートを実装することを諦めてしまった。このように、自分の知識不足が随所に見られ、妥協した部分も非常に多い、とても悔しい結果になってしまった。また、今回の反省点としては、ソースコードの可読性が非常に悪いものになってしまった点である。関数の命名規則に統一性がなく、また、関数内の変数名も統一性がない。また、厳密に構造化出来てなかったとも感じる。関数を分けるべき点で分けず、冗長な関数が出来上がっていた。逆に、使用頻度の低い関数も存在しており（printMoney など）、これは分けるべきではなかった。これらについては今後の課題として、改善していきたい。しかし、その分、自分の力でプログラムを作り上げたという達成感があった。また、今回の演習を通して、C 言語の基礎的な知識を深めることができた。特に理解を深められた点として、ポインタである。はじめはポインタに非常に抵抗感を持っていたが、だんだん慣れてくると、直感的にメモリにアクセスしている感覚がみについて、使いこなせるようになってきた。また、連結リストの実装にチャレンジしたことも非常にいい経験となった。連結リストは、今までの演習ではあまり触れることがなかったので、今回の演習を通して、連結リストの基本的な構造を理解することができた。また、データ構造とアルゴリズムにより興味を持つことができた。今後は、これからもコンピューターサイエンスの勉強を続け、より高度で複雑なプログラムを作成できるようになりたい。

4 ソースコード

```
  //コンビニのレジシステムを再現する
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

typedef struct Item {//商品情報
    char name[100];
    int category;
    int price;
    int purchased;
    struct Item *next; // 次のへのポインタItem 連結リスト//
} Item;

Item *head = NULL; // 連結リストの先頭を指すポインタ

void loadItemsFromFile(const char *filename){
    FILE *file = fopen(filename, "r");
    if (file == NULL) {
        perror("ファイルを開けません");
        return;
    }
    Item *current = head;
    while (!feof(file)) {
        Item *newItem = (Item *)malloc(sizeof(Item));
        if (fscanf(file, "%s␣%d␣%d", newItem->name, &newItem->category, &newItem->price) == 3) {
            newItem->next = NULL;
            newItem->purchased = 0;
            if (current == NULL) {
                head = newItem;
                current = newItem;
            } else {
                current->next = newItem;
                current = newItem;
            }
        } else {
            free(newItem);
            break;
        }
    }
    fclose(file);
}

typedef struct {//紙幣、硬貨の情報
    int num_10000;//円札の枚数10000
    int num_5000;//円札の枚数5000
    int num_2000;//円札の枚数2000
    int num_1000;//円札の枚数1000
    int num_500;//円玉の枚数500
    int num_100;//円玉の枚数100
    int num_50;//円玉の枚数50
    int num_10;//円玉の枚数10
    int num_5;//円玉の枚数5
    int num_1;//円玉の枚数1
} Money;

Money inside_money = {0,10,0,10,30,30,30,30,30,30}; // レジ内にある紙幣、硬貨の情報

int money_sum(Money money){//お金の合計金額を計算
    return money.num_10000 * 10000 + money.num_5000 * 5000 + money.num_2000 * 2000 + money.num_1000 * 1000 + money.num_500 * 500 + money.num_100 * 100 + money.
}

void addMoney(Money *money1, Money *money2){//お金の加算
    money1->num_10000 += money2->num_10000;
```

```
65      money1->num_5000 += money2->num_5000;
66      money1->num_2000 += money2->num_2000;
67      money1->num_1000 += money2->num_1000;
68      money1->num_500 += money2->num_500;
69      money1->num_100 += money2->num_100;
70      money1->num_50 += money2->num_50;
71      money1->num_10 += money2->num_10;
72      money1->num_5 += money2->num_5;
73      money1->num_1 += money2->num_1;
74  }
75
76  void subMoney(Money *money1, Money *money2){//お金の引き算
77      money1->num_10000 -= money2->num_10000;
78      money1->num_5000 -= money2->num_5000;
79      money1->num_2000 -= money2->num_2000;
80      money1->num_1000 -= money2->num_1000;
81      money1->num_500 -= money2->num_500;
82      money1->num_100 -= money2->num_100;
83      money1->num_50 -= money2->num_50;
84      money1->num_10 -= money2->num_10;
85      money1->num_5 -= money2->num_5;
86      money1->num_1 -= money2->num_1;
87  }
88
89  void inside_money_check(Money *inside_money) {//レジ内のお金の枚数を確認 レジ内のお金の枚数が少ない場合は、コンビニ事務所の金庫から補充する //
90      if(inside_money->num_10000 < 0) {
91          inside_money->num_10000++;//この１万円はコンビニ事務所の金庫にある
92      }else if(inside_money->num_5000<5){
93          inside_money->num_5000 += inside_money->num_10000 * 2;
94          inside_money->num_10000 = 0;
95      //else if(inside_money->num.2000<5) 円札は使わない//2000
96      }else if(inside_money->num_1000<5){
97          inside_money->num_1000 += inside_money->num_5000 * 2;
98          inside_money->num_5000 = 0;
99      }else if(inside_money->num_500<5){
100         inside_money->num_500 += inside_money->num_1000 * 5;
101         inside_money->num_1000 = 0;
102     }else if(inside_money->num_100<10){
103         inside_money->num_100 += inside_money->num_500 * 2;
104         inside_money->num_500 = 0;
105     }else if(inside_money->num_50<10){
106         inside_money->num_50 += inside_money->num_100 * 5;
107         inside_money->num_100 = 0;
108     }else if(inside_money->num_10<10){
109         inside_money->num_10 += inside_money->num_50 * 5;
110         inside_money->num_50 = 0;
111     }else if(inside_money->num_5<10){
112         inside_money->num_5 += inside_money->num_10 * 2;
113         inside_money->num_10 = 0;
114     }else if(inside_money->num_1<10){
115         inside_money->num_1 += inside_money->num_5 * 5;
116         inside_money->num_5 = 0;
117     }
118 }
119
120 void printMoney(Money money) { //お金情報の表示これ要る？//
121     printf("万円札の枚数1:%d\n", money.num_10000);
122     printf("千円札の枚数5:%d\n", money.num_5000);
123     printf("千円札の枚数2:%d\n", money.num_2000);
124     printf("千円札の枚数1:%d\n", money.num_1000);
125     printf("円玉の枚数500:%d\n", money.num_500);
126     printf("円玉の枚数100:%d\n", money.num_100);
127     printf("円玉の枚数50:%d\n", money.num_50);
128     printf("円玉の枚数10:%d\n", money.num_10);
129     printf("円玉の枚数5:%d\n", money.num_5);
130     printf("円玉の枚数1:%d\n", money.num_1);
```

```
131 }
132
133 void calculateChange(int money, Money *inside_money) {//お釣り計算
134     Money change = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
135
136     while(money >= 10000){
137         change.num_10000++;
138         money -= 10000;
139     }
140     while(money >= 5000){
141         change.num_5000++;
142         money -= 5000;
143     }
144     /*while(money >= 2000){
145         change.num_2000++;
146         money -= 2000;
147     } */ //円札は使わない2000
148     while(money >= 1000){
149         change.num_1000++;
150         money -= 1000;
151     }
152     while(money >= 500){
153         change.num_500++;
154         money -= 500;
155     }
156     while(money >= 100){
157         change.num_100++;
158         money -= 100;
159     }
160     while(money >= 50){
161         change.num_50++;
162         money -= 50;
163     }
164     while(money >= 10){
165         change.num_10++;
166         money -= 10;
167     }
168     while(money >= 5){
169         change.num_5++;
170         money -= 5;
171     }
172     while(money >= 1){
173         change.num_1++;
174         money -= 1;
175     }
176     subMoney(inside_money, &change);
177     printMoney(change);
178 }
179
180 void register_items(int *total_price, int *tax_price) {// 商品登録
181     int is_continue = 1;
182     char item_name[100];
183     int tax_price_item = 0;
184     while(is_continue){
185         printf("商品名を入力してください\n");
186         scanf("%s", item_name);
187
188         Item *current = head;
189         while(current != NULL && strcmp(current->name, item_name) != 0) {
190             current = current->next;
191         }
192
193         if (current == NULL) {
194             printf("商品が見つかりませんでした。\n");
195         } else {
196             current->purchased = 1;
```

```
197             if(current->category == 0){
198                 tax_price_item += current->price * 0.08 / 1.08; // 消費税（軽減税率）
199             }else{
200                 tax_price_item += current->price * 0.1 / 1.1; // 消費税
201             }
202             *total_price = *total_price + current->price -tax_price_item;
203             *tax_price += tax_price_item;
204             printf("登録商品名: %s\n", current->name);
205             printf("登録商品価格: %d\n", current->price);
206             printf("うち消費税: %d\n", tax_price_item);
207         }
208         printf("登録を続けますか？続ける(1:), 終了0:)\n");
209         scanf("%d", &is_continue);
210     }
211 }
212
213 void receive_payment(Money *received_money) {// 受け取った金額の入力
214     printf("受け取り金額\n");
215     printf("万円札の枚数1:");
216     scanf("%d", &received_money->num_10000);
217     printf("千円札の枚数5:");
218     scanf("%d", &received_money->num_5000);
219     printf("千円札の枚数2:");
220     scanf("%d", &received_money->num_2000);
221     printf("千円札の枚数1:");
222     scanf("%d", &received_money->num_1000);
223     printf("円玉の枚数500:");
224     scanf("%d", &received_money->num_500);
225     printf("円玉の枚数100:");
226     scanf("%d", &received_money->num_100);
227     printf("円玉の枚数50:");
228     scanf("%d", &received_money->num_50);
229     printf("円玉の枚数10:");
230     scanf("%d", &received_money->num_10);
231     printf("円玉の枚数5:");
232     scanf("%d", &received_money->num_5);
233     printf("円玉の枚数1:");
234     scanf("%d", &received_money->num_1);
235 }
236
237 void printReceiptToFile(Item *purchasedItems, int total_price, int tax_price, int received_money, int change) {// レシートの出力
238  // 現在の日時を取得
239     time_t t = time(NULL);
240     struct tm *local = localtime(&t);
241
242     // ファイル名を格納するための文字列バッファ
243     char filename[50];
244
245     // ファイル名をを使ってフォーマットsprintf
246     sprintf(filename, "%d-%d-%d-%d:%d-receipt.txt", local->tm_year + 1900, local->tm_mon + 1, local->tm_mday, local->tm_hour, local->tm_min);
247
248     // ファイルを開く
249     FILE *file = fopen(filename, "w");
250     if (file == NULL) {
251         perror("ファイルを開けません");
252         return;
253     }
254     fprintf(file, "---レシート---\n");
255     fprintf(file, "%d-%d-%d %d:%d\n", local->tm_year + 1900, local->tm_mon + 1, local->tm_mday, local->tm_hour, local->tm_min); // 現
在の日付と時刻
256
257     fprintf(file, "\購入商品リストn:\n");
258     while (purchasedItems != NULL) {
259         if(purchasedItems->purchased == 1){
260         fprintf(file, "商品名: %s, 価格: %円d\n", purchasedItems->name, purchasedItems->price);
261         }
262         purchasedItems = purchasedItems->next;
```

```
263     }
264
265     fprintf(file, "\合計金額n:␣%円d\n", total_price);
266     fprintf(file, "消費税:␣%円d\n", tax_price);
267     fprintf(file, "お預かり金額:␣%円d\n", received_money);
268     fprintf(file, "お釣り:␣%円d\n", change);
269     fprintf(file, "-------------------\n");
270
271     fclose(file);
272 }
273
274 void calculate(int *total_price, int *tax_price) {// 会計
275     Money received_money = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
276     int current_total_price = 0;
277     int current_tax_price = 0;
278
279     register_items(&current_total_price, &current_tax_price);
280
281     printf("合計金額:%d\n", current_total_price+current_tax_price);
282     receive_payment(&received_money);
283
284
285     printf("お預かり金額:%d\n", money_sum(received_money));
286
287     if(money_sum(received_money) < current_total_price) {
288         printf("お預かり金額が足りません。\n");
289         return;
290     }
291     addMoney(&inside_money, &received_money);
292     int change = money_sum(received_money) - current_total_price;
293     printf("お釣り:%d\n", change);
294     calculateChange(change, &inside_money);
295     printReceiptToFile(head, current_total_price, current_tax_price, money_sum(received_money), change);
296     *total_price += current_total_price;
297     *tax_price += current_tax_price;
298 }
299
300 void check(int total_price_today,int initial_money,int total_tax_today){//在高点検
301     printf("在高点検を行います。\n");
302     printf("レジ内の金額:%d\n", money_sum(inside_money));
303     printf("本日の売上金額:%d\n", total_price_today-total_tax_today);
304     printf("本日の消費税額:%d\n", total_tax_today);
305     printf("差分+-:%d\n", money_sum(inside_money) - (initial_money + total_price_today+total_tax_today));
306 }
307
308 void settlement(int *total_price_today, int *total_tax_today, int *refund_today, int *public_survice_today){//精算業務
309     printf("精算を行います。\n");
310     printf("本日の返金額:%d\n", *refund_today);
311     *refund_today = 0;
312     printf("本日の収納代行件数:%d\n", *public_survice_today);
313     *public_survice_today = 0;
314     int sum =  *total_price_today;
315     calculateChange(sum,&inside_money);
316      printf("本日の売上金額:%d\n", *total_price_today);
317     *total_price_today = 0;
318     printf("本日の消費税額:%d\n", *total_tax_today);
319     *total_tax_today = 0;
320     printf("レジ内の金額:%d\n", money_sum(inside_money));
321 }
322
323 void refund(int *total_price_today, int *total_tax_today, int *refund_today, Money *inside_money) {// 返金
324     char item_name[100];
325     int refund_amount;
326
327     printf("返品する商品名を入力してください\n");
328     scanf("%s", item_name);
```

```
329
330     Item *current = head;
331     while(current != NULL && strcmp(current->name, item_name) != 0) {
332         current = current->next;
333     }
334
335     if (current == NULL) {
336         printf("商品が見つかりませんでした。\n");
337     } else {
338         printf("返品商品名:␣%s\n", current->name);
339         printf("返品商品価格:␣%d\n", current->price);
340
341         refund_amount = current->price;
342         printf("返金額:␣%d\n", refund_amount);
343         calculateChange(refund_amount, inside_money); // レジ内のお金を更新
344         *total_price_today -= refund_amount; // 売上金額から返金額を減算
345
346         if(current->category == 0) {
347             *total_tax_today -= refund_amount * 0.08 / 1.08; // 消費税から返金額の消費税分を減算
348         } else {
349             *total_tax_today -= refund_amount * 0.1 / 1.1; // 消費税から返金額の消費税分を減算
350         }
351         *refund_today += refund_amount; // 返金総額に返金額を加算
352     }
353 }
354
355 int main(){
356     int initial_money = money_sum(inside_money); // レジの初期金額
357     int mode; // モード
358     int total_price_today = 0; // 本日の売上金額
359     int total_tax_today = 0; // 本日の消費税額
360     int refund_today = 0; // 本日の返金額
361     int public_survice_today = 0; // 本日の収納代行件数
362     int is_continue = 1; // 続けるかどうか
363     loadItemsFromFile("items.txt"); // 商品情報の読み込み
364
365     while(is_continue){
366         printf("モードを選択してください\n");
367         printf("会計1:\n");
368         printf("レジ点検2:\n");
369         printf("レジ精算3:\n");
370         printf("返金4:\n");
371         printf("収納代行5:\n");//収納代行は、コンビニのレジで公共料金の支払いなどを行うこと
372         printf("レジ休止6:\n");
373         printf("入力:");
374         scanf("%d", &mode);
375
376         switch(mode) {
377         case 1:
378             calculate(&total_price_today, &total_tax_today);
379             break;
380         case 2:
381             printf("レジ点検\n");
382             printf("在高点検\n");
383             check(total_price_today, initial_money, total_tax_today);
384             break;
385         case 3:
386             printf("レジ精算\n");
387             printf("精算します。\n");
388             settlement(&total_price_today, &total_tax_today, &refund_today, &public_survice_today);
389             break;
390
391         case 4:
392             printf("返金\n");
393             refund(&total_price_today, &total_tax_today, &refund_today, &inside_money);
394             break;
```

```
395             case 5:
396                 printf("収納代行\n");
397                 public_survice_today++;//今回は収納代行件数をカウントするだけで、実際に実装はしない
398                 printf("収納代行サービスを行いました。\n"    );
399                 break;
400             case 6:
401                 printf("レジ休止\n");
402                 printf("休止します。\n");
403                 is_continue = 0;
404                 break;
405             default:
406                 printf("モードが不正です\n");
407             }
408             inside_money_check(&inside_money);
409         }
410 }
```

5　実行結果

##会計
モードを選択してください
1: 会計
2: レジ点検
3: レジ精算
4: 返金
5: 収納代行
6: レジ休止
入力:1
商品名を入力してください
おにぎり
登録商品名: おにぎり
登録商品価格: 120
うち消費税: 8
登録を続けますか？(1: 続ける, 0: 終了)
1
商品名を入力してください
セブンスター
登録商品名: セブンスター
登録商品価格: 600
うち消費税: 62
登録を続けますか？(1: 続ける, 0: 終了)
1
商品名を入力してください
クランキーチキン
登録商品名: クランキーチキン
登録商品価格: 291
うち消費税: 83
登録を続けますか？(1: 続ける, 0: 終了)
0
合計金額:1011
受け取り金額
1 万円札の枚数:0
5 千円札の枚数:1
2 千円札の枚数:0
1 千円札の枚数:0
500 円玉の枚数:0
100 円玉の枚数:0
50 円玉の枚数:0
10 円玉の枚数:0
5 円玉の枚数:0
1 円玉の枚数:0
お預かり金額:5000
お釣り:4142

1 万円札の枚数:0
5 千円札の枚数:0
2 千円札の枚数:0
1 千円札の枚数:4
500 円玉の枚数:0
100 円玉の枚数:1
50 円玉の枚数:0
10 円玉の枚数:4
5 円玉の枚数:0
1 円玉の枚数:2

##返金
モードを選択してください
1: 会計
2: レジ点検
3: レジ精算
4: 返金
5: 収納代行
6: レジ休止
入力:4
返金
返品する商品名を入力してください
クランキーチキン
返品商品名: クランキーチキン
返品商品価格: 291
返金額: 291
1 万円札の枚数:0
5 千円札の枚数:0
2 千円札の枚数:0
1 千円札の枚数:0
500 円玉の枚数:0
100 円玉の枚数:2
50 円玉の枚数:1
10 円玉の枚数:4
5 円玉の枚数:0
1 円玉の枚数:1

##レジ精算
レジ精算
精算します。
精算を行います。
本日の返金額:291
本日の収納代行件数:0
1 万円札の枚数:0
5 千円札の枚数:0
2 千円札の枚数:0
1 千円札の枚数:0
500 円玉の枚数:1
100 円玉の枚数:0
50 円玉の枚数:1
10 円玉の枚数:1
5 円玉の枚数:1
1 円玉の枚数:2
本日の売上金額:567
本日の消費税額:131
レジ内の金額:79980
モードを選択してください
1: 会計
2: レジ点検
3: レジ精算
4: 返金
5: 収納代行
6: レジ休止

##レジ点検
入力:2

レジ点検
在高点検
在高点検を行います。
レジ内の金額:79980
本日の売上金額:0
本日の消費税額:0
+-差分:0
モードを選択してください
1: 会計
2: レジ点検
3: レジ精算
4: 返金
5: 収納代行
6: レジ休止

##入力ファイルの例
セブンスター 1 600
テリアメンソール 1 580
おにぎり 0 120
ペットボトルのお茶 0 150
パン 0 200
ソフトクリームバニラ 0 270
香るベトナムカカオチョコソフト 0 313
プレミアムショコラソフト 0 421
ダブル蜜いもソフト 0 464
なめらかプリンパフェ 0 410
ピスタチオカカオパフェ 0 432
クランキーチキン 0 291
X フライドポテト 0 291
北海道チーズインナゲット 0 291

##出力ファイルの例
--- レシート ---
2023-12-21 12:29

購入商品リスト:
商品名: セブンスター, 価格: 600 円
商品名: おにぎり, 価格: 120 円

合計金額: 604 円
消費税: 116 円
お預かり金額: 5000 円
お釣り: 4396 円
