بسمه تعالی



عنوان:

**گزارشکار تمرین دوم**

**«طراحی سیستم بانک»**

استاد:

**مهندس نرگس بطحائیان**

دانشجو:

**نسترن منصوری**

**9612358041**

ترم تحصیلی:

**4001**

**فهرست**

[**هدف: 4**](#_Toc92947601)

[**مقدمه: 4**](#_Toc92947602)

[**کلاس Date: 5**](#_Toc92947603)

[**Date.h: 5**](#_Toc92947604)

[**Date.cpp: 6**](#_Toc92947605)

[**کانستراکتور 6**](#_Toc92947606)

[**تابع set\_year: 7**](#_Toc92947607)

[**تابع get\_year: 7**](#_Toc92947608)

[**تابع print: 8**](#_Toc92947609)

[**کلاس Client: 9**](#_Toc92947610)

[**Client.h: 9**](#_Toc92947611)

[**Client.cpp 10**](#_Toc92947612)

[**تابع createAccount: 10**](#_Toc92947613)

[**تابع set\_username 11**](#_Toc92947614)

[**تابع get\_username 11**](#_Toc92947615)

[**تابع set\_ip 12**](#_Toc92947616)

[**تابع get\_ip 12**](#_Toc92947617)

[**تابع set\_cardNumber 13**](#_Toc92947618)

[**تابع get\_cardNumber 13**](#_Toc92947619)

[**تابع set\_inventory 13**](#_Toc92947620)

[**تابع get\_inventory 14**](#_Toc92947621)

[**تابع deposit 14**](#_Toc92947622)

[**تابع withdraw 14**](#_Toc92947623)

[**تابع set\_createAccountDate 15**](#_Toc92947624)

[**تابع set\_acounExpirationAuto 15**](#_Toc92947625)

[**تابع set\_acounExpirationManual 15**](#_Toc92947626)

[**تابع print 16**](#_Toc92947627)

[**Main برنامه: 16**](#_Toc92947628)

[**تابع commands: 18**](#_Toc92947629)

[**تابع add\_acount: 19**](#_Toc92947630)

[**تابع renew: 20**](#_Toc92947631)

[**تابع invent: 21**](#_Toc92947632)

[**تابع transfer: 22**](#_Toc92947633)

# هدف:

هدف از پیاده سازی این تمرین طراحی و پیاده سازی سیستم بانکی است.

# مقدمه:

کد پیاده سازی شده برای این تمرین به طور کلی از سه قسمت تشکیل شده است. دو کلاس و main برنامه که د رادامه به طور مفصل توضیح داده می شود.

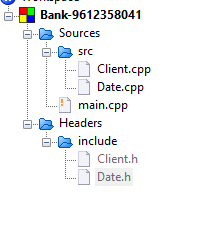
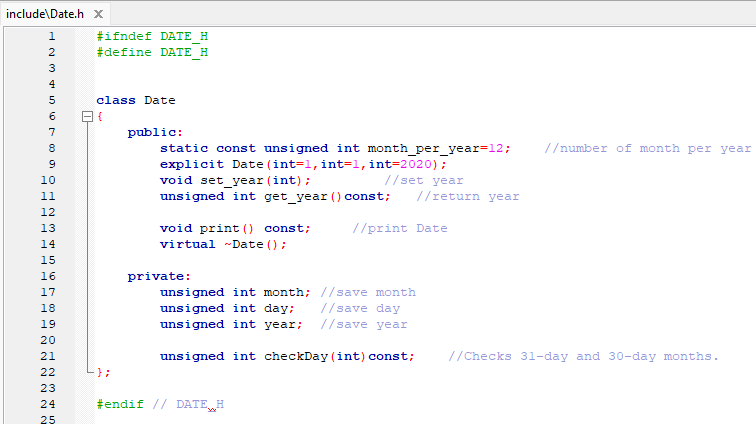


Figure 1:اجزا پیاده سازی شده

# کلاس Date:

این کلاس برای ذخیره تواریخ موجود در سیستم مورد استفاده قرار میگیرد که از دو بخش .h و .cpp تشکیل شده است.

## Date.h:

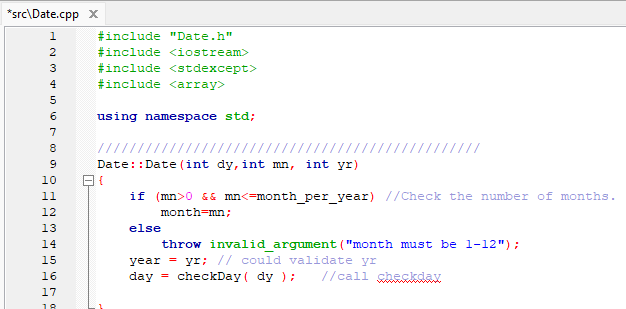


همانطر که در تصویر مشخص است، در بخش private کلاس سه متغیر از نوع unsigned int وجود دارد که به ترتیب برای ذخیره سازی ماه، روز و سال هستند. یک تابع هم وجود دارد به نام checkday که به عنوان ورودی یک متغیر از نوع integer می گیرد و برای بررسی 31 یا 30 روزه بودن ماه های سال است.

در بخش public متغیر ثابتی تعریف شده است از نوع integer به نام month\_oper\_year که نشان دهنده ی تعداد ماه های یک سال است. در خط بعدی کاسنتراکتور کلاس وجود دارد که مقادیر پیش فرض 1/1/2020 را در متغیرهای خصوصی قرار می دهد.

در خط بعد تابع set\_year وجود دارد برای تغییر متغیر year که در ادامه دلیل پیاده سازی آن را متوجه خواهید شد و د رنهایت تابع get\_year که مقدار موجود در متغیر year را برمی گرداند.

## Date.cpp:

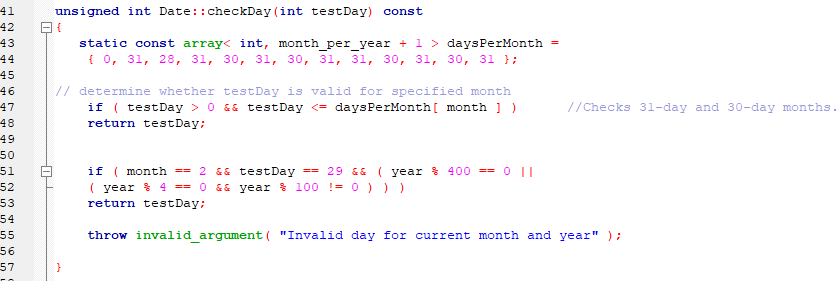


کتابخانه های استفاده شده در این کلاس همانطور که مشخص است به جز iostream، کتابخانه stdexcept می باشد که برای رخ دادن استثناهاست.

کانستراکتور:

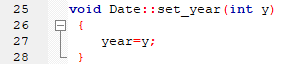
کانستاکتور این کلاس با دریافت سه متغیر صحیح به نام mn,dy,yr کا رخود را آغاز می کند. ابتدا چک میکند که مقدار متغیر mn از 0 بزرگتر بوده و از month\_per\_year کوچکتر باشد که در صورت داشتن این شرط آن را در متغیر month قرار می دهد در غیر اینصورت یک استثنا پرتاب می کند. سپسyr را در year قرار داده و برای قرار دادن متغیر dy در day تابع checkday را فراخوانی می کند.

تابع checkday:



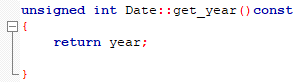
این تابع به عنوان ورودی یک متغیر صحیح میگیرد به نام testDay. سپس با تعریف آرایه ای از نوع int به طول month\_per\_year+1 به نام daysPerMonth و ان را با مقادیری که در تصویر مشخص است مقداردهی می کند. در خط 47 چک می شود که اگر ورودی تابع از 0 بزرگتر و از تعداد روزهای ماه مورد نظر کمتر بود؛ مقدار ورودی را برمی گرداند. اما اگر ماه دوم باشیم و مقدار ورودی 29 باشد و سال ورودی کبیسه باشد د راینصورت هم مقدار ورودی مرد پذیرش است و برگردانده می شود. در غیر اینصورت یک استثنا پرتاب خواهد شد.

### تابع set\_year:



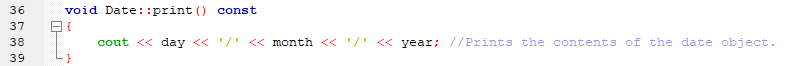
این تابع مقدار ورودی y را در year ذخیره می کند.

### تابع get\_year:



این تابع هم مقدار موجود در متغیر year را برمی گرداند.

### تابع print:

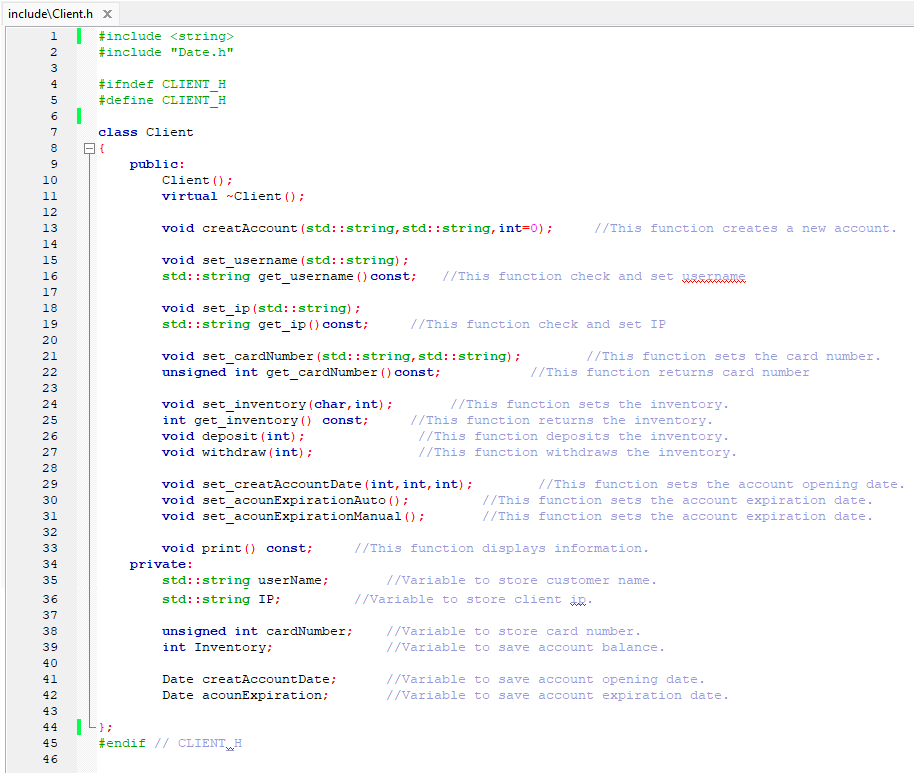


این تابع تاریخ ورودی را چاپ می کند.

# کلاس Client:

این کلاس برای ذخیره اطلاعات مشتریان بانک در سیستم مورد استفاده قرار میگیرد که از دو بخش .h و .cpp تشکیل شده است.

## Client.h:

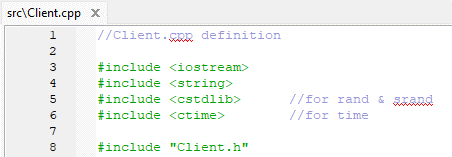


در بخش private دو متغیر از نوع رشته وجود دارد برای ذخیره username و IP ها. یک متغیر صحیح مثبت برای ذخیره شماره کارت. یک متغیر صحیح برای ذخیره موجودی و دو متغیر از نوع کلاس تاریخ برایذخیره تاریخ افتتاح حسب و انقضا کارت.

در بخش public:

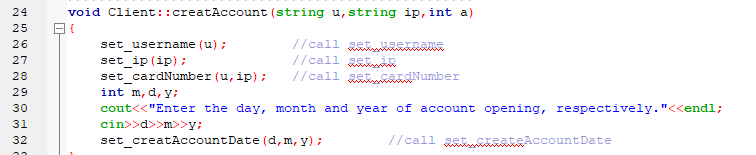
* تابعی برای ایجاد حساب
* تابعی برای ثبت وبرگرداندن username
* تابعی برای ثبت و برگرداندن IPها
* تابعی برای ثبت و برگرداندن شماره کارت
* تابعی برای تغییر موجودی،افزایش و کاهش
* تابعی برای ثبت تاریخ افتتاح حساب
* تابعی برای چاپ اطلاعات

Client.cpp:



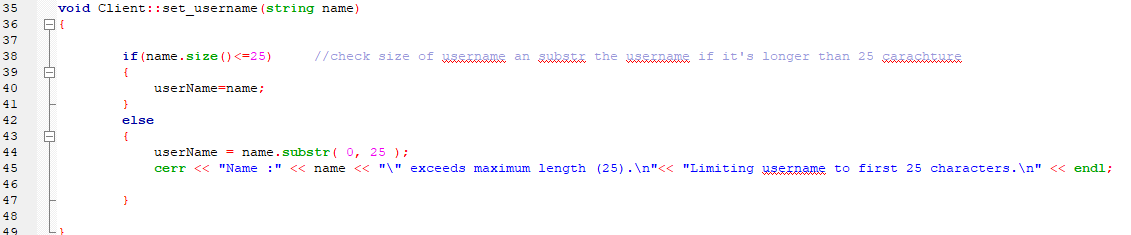
در این بخش از کلاس مشتری از کتابخانه های iostream, string, cstdlib و ctime(برای زمان) استفاده شده است.

تابع createAccount:



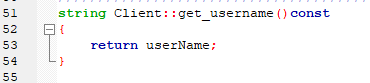
این تابع دو متغیر از نوع رشته و یک متغیر از نوع integer را به عنوان ورودی دریافت می کند. سپس با هر یک از پارامترهای ورودی یک تابع را فراخوانی می کند.(توابع Set\_ip و set\_username و set\_cardNumber) سپس با دریافت روز و ماه و سال افتتاح از کاربر تابع Set\_creatAccountDate را فراخوانی می کند. پارامترهای اول دوم به عنوان username و IP و پارامتر سوم مشخص کننده موجودی کاربر است که در هدر به صورت پیش فرض مقدار آن 0 قرار داده شد.

تابع set\_username:



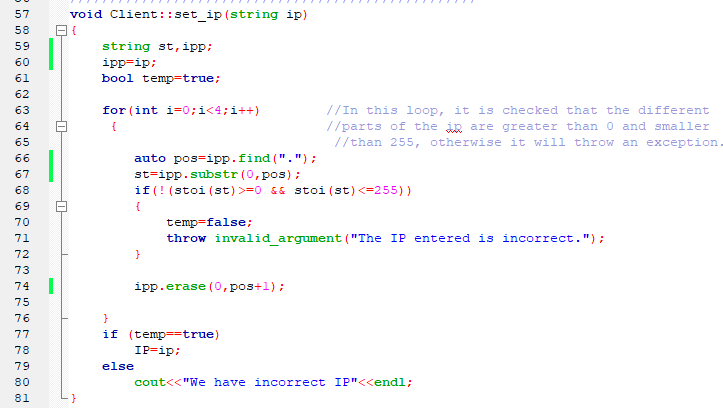
این تابع تنها یک پارامتر ورودی دارد از نوع رشته. که این پارامتر را بررسی می کند که مقدار آن از 25 کاراکتر بیشتر نباشد. اگر بیش از 25 کاراکتر بود تنها 25 تای اول آن را به عنوان نام ذخیره می کند و این موضوع را به کاربر اطلاع می دهد.

تابع get\_username:



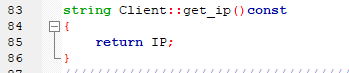
این تابع هم نام ذخیره شده را برمی گرداند.

تابع set\_ip:



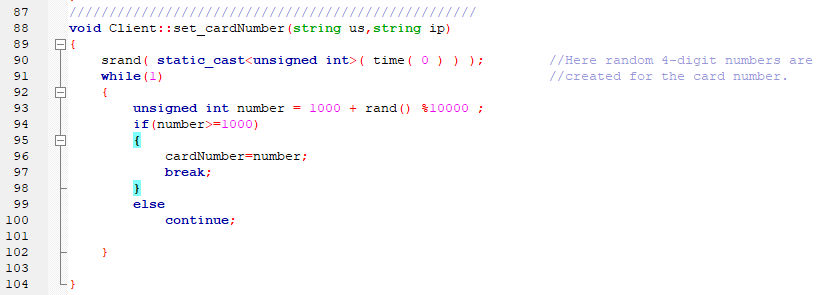
این تابع پارامتر ورودی خود را که از نوع رشته است بعد از دریافت به 4 بخش تقسیم می کند(به این دلیل که هر IP از 4 بخش تشکیل می شود.)سپس هر یک از این 4 بخش را به عدد صحیح تبدیل می کند و بررسی میکند که مقدار آن ا ز0 بیشتر یا مساوی و از 255 کوچکتر یا مساوی باشد. اگر قسمتی از این پارامتر در این بازه نباشد یک استثنا پرتاب می شود.

تابع get\_ip:



این تابع مقدار ip ذخیره شده را بر می گرداند.

تابع set\_cardNumber:



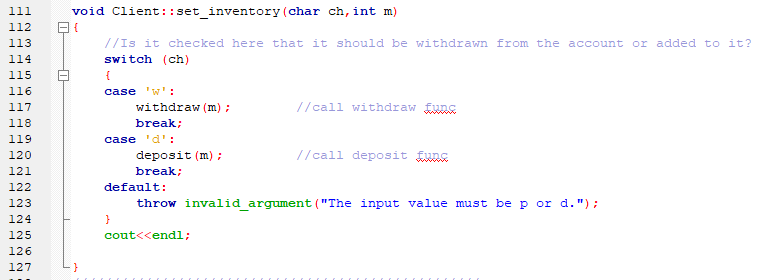
در این تابع به طور رندوم یک عدد 4 رقمی ایجاد می شود. و در متغیر cardNumber ذخیره می گردد.

تابع get\_cardNumber:



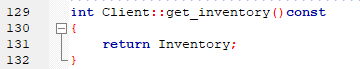
این تابع مقدار موجود در متغیر cardNumber را بر می گرداند.

تابع set\_inventory:



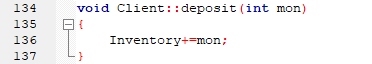
این تابع دو پارامتر ورودی دارد.یکی نشان دهنده نوع برداشت یا واریز و دیگری برای مشخص کردن مقدار برداشتی یا واریزی می باشد. در این تابع ابتدا بررسی می شود که مقدار کاراکتر ورودی چیست؟ اگر 'w' بود تابع withdraw فراخوانی می شود و اگر 'd' بود تابع deposit فراخوانی می شود.

تابع get\_inventory:



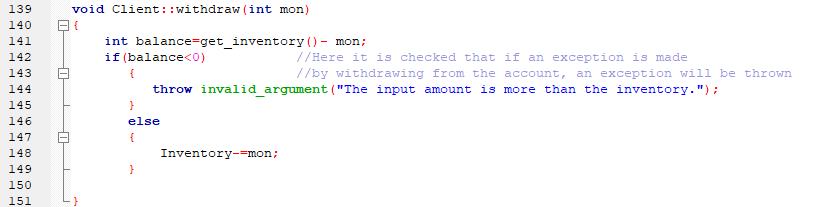
این تابع هم مقدار موجودی را برمی گرداند.

تابع deposit:



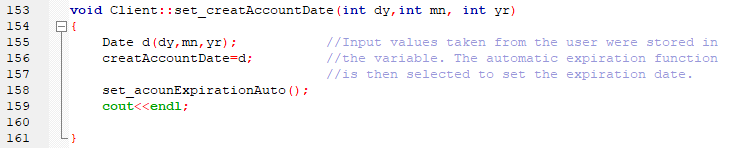
این تابع مقدار پارامتر ورودی را به مقدار موجودی مشتری اضافه می کند.

تابع withdraw:



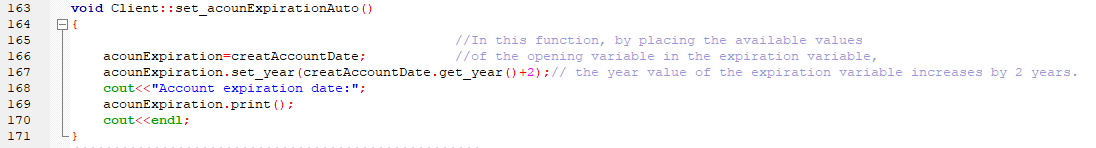
این تابع بعد از دریافت پارامتر ورودی بررسی می کند که آیا با کم کردن این مقدار از موجودی مشتری مقدار موجود منفی خواهد بود یا خیر؟ اگر منفی باشد یک استثنا پرتاب خواهد کرد درغیر اینصورت مقدار ورودی را از موجودی کم می کند.

تابع set\_createAccountDate:



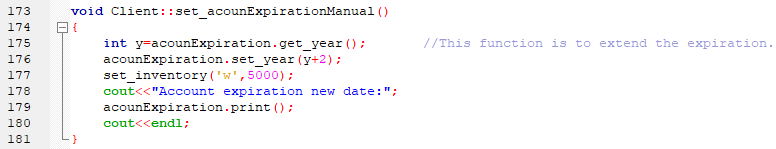
این تابع ابتدا یک شی با ورودی هایی که می گیرد می سازد، سپس محتوای شی ساخته شده در متغیر creatAccountDate قرار می دهد. بعد تابع set\_acounExpirationAuto را فراخوانی می کند.

تابع set\_acounExpirationAuto:



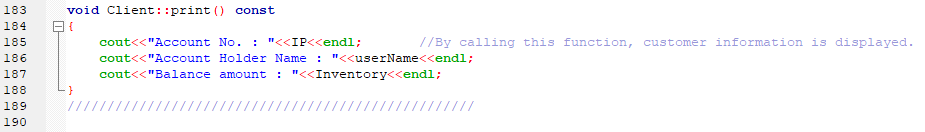
این تابع محتوای موجود در متغیر creatAccountDate را در متغیر accounExpiration ذخیره می کند. سپس با فراخوانی تابع set\_year و get\_year از کلاس Date مقدار سال را دوسال افزایش می دهد. و بعد تاریخ انقضا را چاپ می کند.

تابع set\_acounExpirationManual:



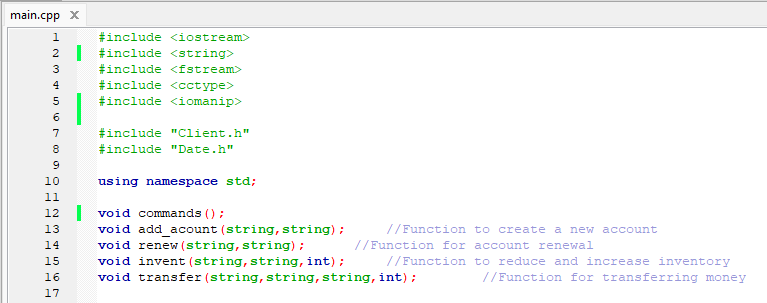
این تابع ابتدا سال تاریخ انقضا را در یک متغیری به نام y ذخیره می کند. سپس متغیر سال تاریخ انقضا را 2 سال افزایش می دهد و بعد تابع set\_inventory را فراخوانی می کند تا مبلغ مورد نیاز برای تمدید حساب(5000) از موجودی مشتری کاسته شود.

تابع print:

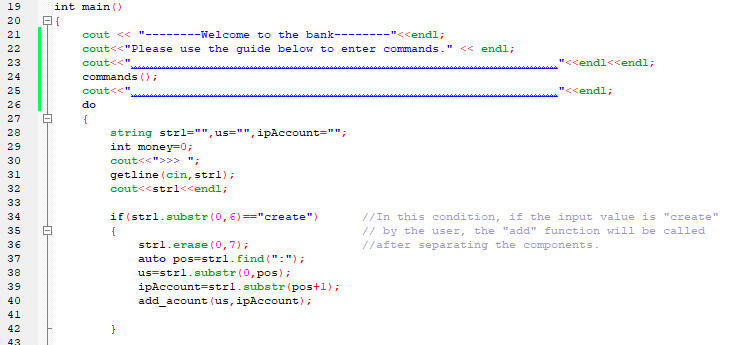


این تابع هم اطلاعات مشتریان را چاپ می کند.

# Main برنامه:

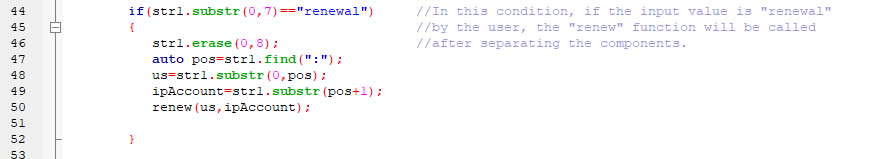


همانطور که در تصویر مشخص است، از کتابخانه های بالا در این قسمت کد استفاده شده است. همچنین هردو کلاس Client و Date هم include شده اند. در ادامه 5 تابع داریم که توضیح داده خواهد شد.

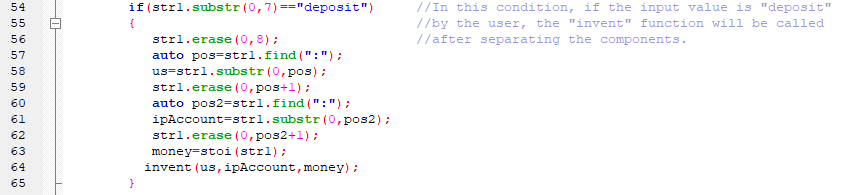


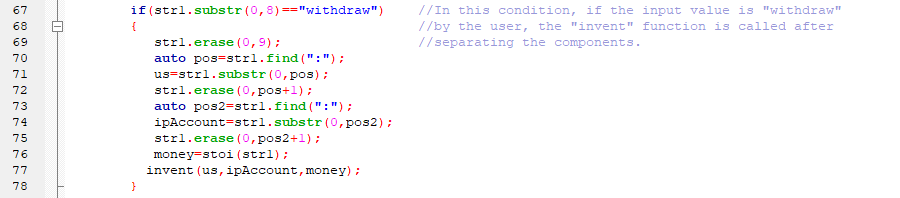
در شروع main بعد از خوش آمد گویی به کاربر و فراخوانی تابع راهنمای دستورات وارد یک حلقه do...while می شود.سه متغیر از نوع رشته تعریف شده است که برای ذخیره ورودی کاربر و username و IP هستند. همچنین یک متغیر دیگر هم از نوع صحیح وجود داردکه مبلغ موردنظر مشتری در آن ذخیره می شود. از کاربر میخواهد تا دستور مورد نظر را وارد کند.

بعد از ورود دستور برنامه شروع به چک کردن دستورات می کند. به این صورت که اگر کله ی اول وارد شده توسط کاربر "create" باشد بعد از جدا سازی username و IP از هم تابع add\_account فراخوانی می شود.

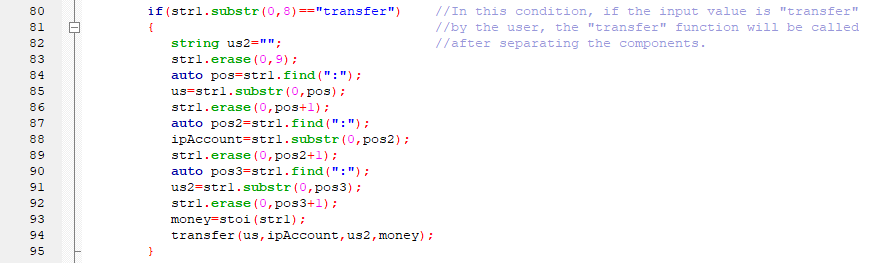


اگر دستور وارد شده با "renewal" شروع شود تابع renew بعد از جداسازی username و IP فراخوانی می شود.

 اگر دستور وارد شده با "deposit" شروع شود تابع invent بعد از جداسازی username و IP فراخوانی می شود.

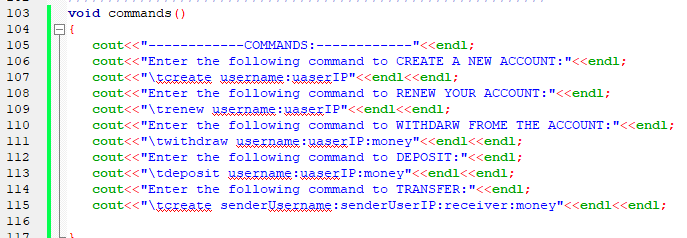


اگر دستور وارد شده با "withdraw" شروع شود تابع invent بعد از جداسازی username و IP فراخوانی می شود.



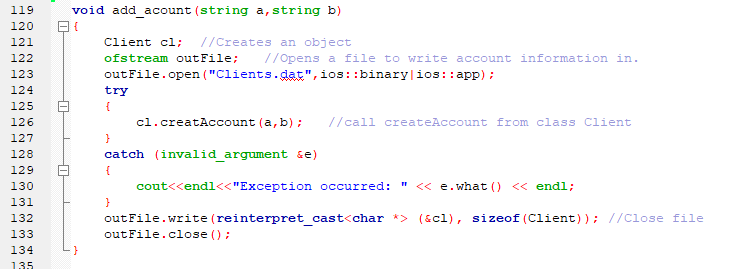
اگر دستور وارد شده با "transfer" شروع شود تابع transfer بعد از جداسازی username و IP فرستده و username گیرنده فراخوانی می شود.

### تابع commands:



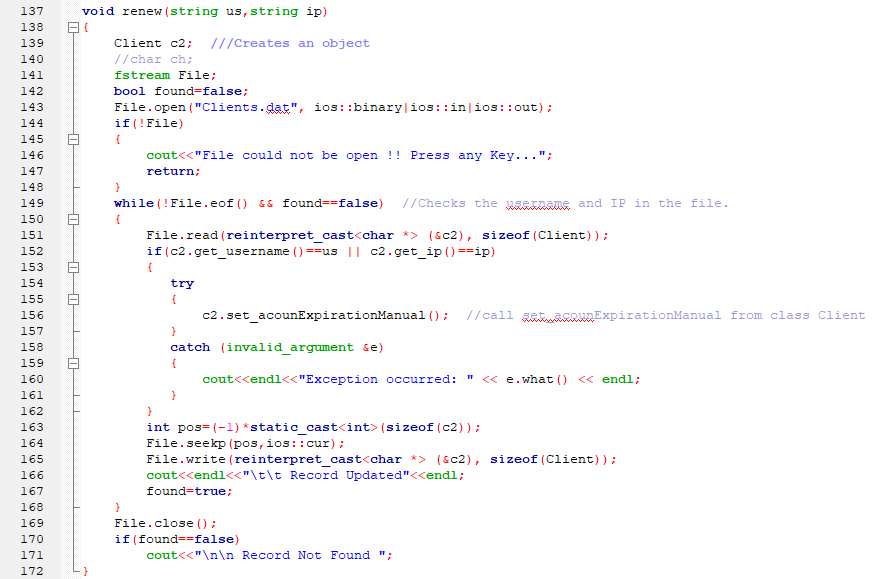
این تابع دستوراتی که کاربر لازم است وارد کند را به عنوان راهنما به او شنان می دهد.

### تابع add\_acount:



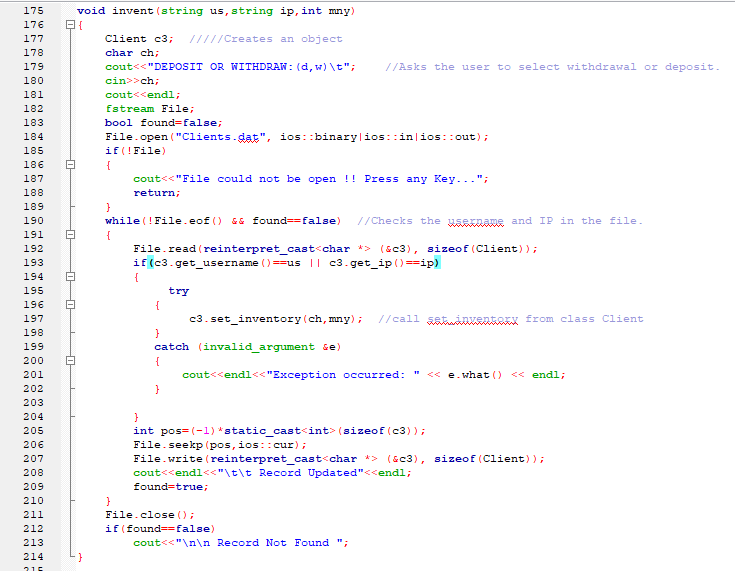
این تابع برای ایجاد حساب جدید فراخوانی می شود.ابتدا یک شی از کلاس مشتری ایجاد میک ند سپس فایلی را به نام Clients باز میکند و اطلاعات مشتری را در آن وارد می کند. این کار با فراخوانی تابع creatAcccount از کلاس مشتری انجام می دهد. در انتها فایل را می بندد.

### تابع renew:



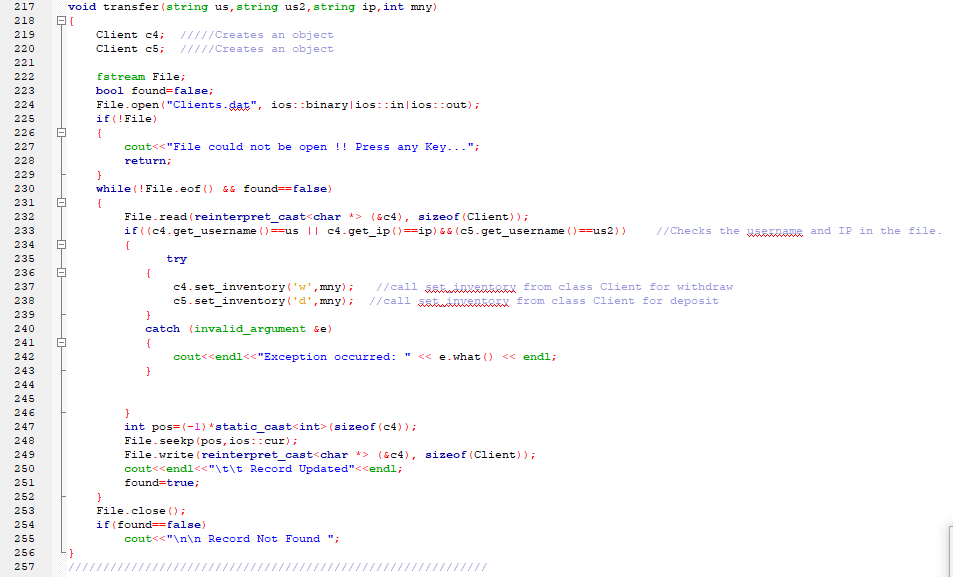
این تابع برای تمدید انقاضا حساب مشتریان فراخوانی می شود.به این وصرت که بعد از ایجاد یک شی از کلاس مشتری و باز کردن فایل Clients در آن به دنبال مشتری می گردد که اطلاعاتش را به عنوان پارامتر ورودی دریافت کرده است. در صورت پیدا کردن مشتری، تابع set\_acountExpirationManual را فراخوانی کرده و تاریخ انقضا را تمدید می کند. اگر نتواند مشتری را پیدا کند پیغام "Record Not Found" را می دهد.

### تابع invent:



این تابع برای برداشت و واریز از حساب مشتری تعریف شده است. به این صورت عمل می کند که بعد از ایجاد یک شی از کلاس مشتری از کاربر میخواهد که مشخص کند میخواهد برداشت انجام دهد یا واریز(با ورود کارکتر d یا w).سپس فایل Clients را باز کرده و در آن به دنبال مشتری با اطلاعات ورودی می گردد. مانند تابع قبل اگر مشتری موردنظر را پیدا کند تابع set\_inventory را با کارکتر ورودی توسط کاربر و مقدار پول فراخوانی می کند. در صورت نیافتن مشتری هم مانند تابع قبل پیغام می دهد.

### تابع transfer:



در این تابع برعکس توابع دیگر اطلاعت دو مشتری گرفته می شود یکی کسی که میخواهد پول را انتقال دهد و دیگری کسی که پول به حسابش وریز خواهد شد. بعد از ایجاد دو شی از کلا سمشتری و باز کردن فایل ابتدا به دنبال اطلاعات فرستده میگردد سپس به دنبال اطلاعات گیرنده به این دلیل که هردو باید در بانک حساب داشته باشند. بعد از یافتن مشتریان موردنظر تابع set\_inventory را با کاراکتر 'w' برای فرستنده فراخوانی می کند تا مقدار مبلغ موردنظر از حساب او کاسته شود. سپس تابع set\_inventory را با کاراکتر 'd' برای گیرنده فراخوانی می کند تا مبلغ موردنظر به حساب او واریز شود.در صورت پیدا نکردن اطلاعت هر یکاز مشتریان در فایل پیغام "Record Not Found" را چاپ می کند.