

Natalia Góras  
+48 733-000-906  
natalikag2004@gmail.com

Temat pracy:

Projekt i implementacja systemu wspomagającego zarządzanie biblioteką.  
Automatyczne składanie zlecenia na zakup książek.

Wymagania dotyczące zadania indywidualnego:

- ✓ Dokumentacja – zawiera sformułowanie zadania, schemat blokowy całego programu (ogólny, na wysokim poziomie szczegółowości), schematy blokowe wybranych, ważnych podprogramów, opis rozwiązania, przykłady działania programu.
- ✓ Program powinien być napisany w sposób strukturalny – podział na podprogramy, w jednym podprogramie powinny być umieszczone wywołania innych podprogramów,
- ✓ Tam, gdzie jest to uzasadnione, możliwe należy korzystać z plików (dane, wyniki należy zapisywać w plikach),
- ✓ W zadaniu rozważamy maksymalny jego zakres, tzn. co można w nim najwięcej zrobić,
- ✓ Obrona – kilka pytań z projektu.

## Spis treści

1)	Informacje ogólne.....	2
2)	Sposób działania programu .....	3
3)	Schemat blokowy .....	5
	Schemat przedstawia funkcję w pliku główny int main() .....	6
	Schemat przedstawia funkcje logowanie_pracownika(): .....	7
4)	Wyjaśnienie pliku głównego: main.cpp.....	9
	Funkcja główna programu: 'int main()' .....	10
	Funkcje pomocnicze: .....	11
	Funkcja związane z kontem pracownika: .....	12
	Funkcja związana z kontem czytelnika:.....	14
5)	Wyjaśnienie pliku nagłówkowego szyfrowanieHasla.h.....	15
	Funkcja logowanie_pracownika() .....	16
6)	Wyjaśnienia do pliku bazaKsiazek.h .....	18
	Funkcja zakup_ksiazki() .....	19
	Funkcja zloz_zamowienie() .....	20
7)	Wyjaśnienie pliku zadaniaNaKsiazkach.h.....	28
	Funkcja domow_ksiazke().....	29
	Funkcja stan_ksiazek() .....	30
8)	Wyjaśnienie pliku znajdzKsiazke.h.....	33
	Funkcja znajdz_ksiazke() .....	35
9)	wyjaśnienie pliku rejestracjaK.h .....	37
	Funkcja rejestracja_klienta().....	38
10)	Wyjaśnienie pliku klient.h .....	43
	Funkcja wypożycz_ksiazke() .....	45
	Funkcja sprawdź_konto_klienta(): .....	46
	Funkcja zwróć_ksiazke(): .....	47

## 1) INFORMACJE OGÓLNE

Moja praca została zaprojektowana w kompilatorze „Code::Blocks 20.03”. To w nim uruchamiane są wszystkie pliki. Projekt jest podzielony na wiele plików, by zachować spójność oraz funkcjonalność programu. Nazwy plików są pisane w sposób logiczny do funkcji w nich zawartych. A to wszystko znajduje się w jednym folderze by odwołując się do plików nie podawać całych ścieżek tylko ich nazwy. Nie zapominam również że program nie

używa tylko plików .cpp oraz .h, ale także .txt, w których znajdują się między innymi informacje o książkach, klientach oraz wypożyczeniach i zwrotach.

Główny plik, z którego uruchamiam cały projekt to: main.cpp, a do niego podłączone są pliki nagłówkowe, w których znajdują się funkcje wywoływane przez plik główny: - bazaKsiazek.h, -klient.h, -rejestracjaK.h, -zadaniaNaKsiazkach.h, -szyfrowanieHasla.h, -znajdzKsiazke.h. By to zrobić muszę przejść do zakładki: File: New: File...: C/C++header: i postępuję zgodnie z wskazówkami, lecz pamiętając by wybrać dobry folder oraz podpisać odpowiednio pliki. W programie wykorzystuję jak już wcześniej wspomniałam pliki tekstowe, które umożliwiają mi zapisywanie w pamięci danych potrzebnych w programie. Są nimi: - wyporzczenieK.txt, -numer\_ksiazki.txt, -numer\_karty.txt, -ksiazki.txt, -klient.txt.

Uwagi:

- niestety w moim programie w wielu miejscach nie ma napisanej litery „I”, ponieważ mój komputer nie ma możliwości poprzez połączenie Alt + I, napisanie tego znaku. W niektórych momentach kopiowałam z Worda literę i wklejałam do kodu, ponieważ błąd jest bardzo widoczny, lecz takich miejsc jest bardzo dużo i sama nie jestem w stanie tego wszystkiego znaleźć.
- przy wpisywaniu cyfr, które odpowiadają za przechodzenie po kodzie i wywoływanie funkcji np. w menu itp., nie wyświetlają się błędy, jednak jeśli wprowadzę literę automatycznie program się zapętla i wyświetla błąd.
- Każda książka w bibliotece ma swoje id, które ułatwia mi pracę związaną z książkami, wypożyczanie, zamawianie itp. Lecz nrKsiazki jest nadawany automatycznie przez program, który pobiera numer z pliku wcześniej zapisanej książki i przechowuje go do momentu zakupu nowej książki, wtedy zachodzi zmiana cyfry w pliku nrKarty.txt. I błąd może wykonać się w momencie gdy użytkownik sam będzie chciał ingerować w plik z książkami. Ponieważ następnym razem gdy będę chciała dokupić książkę nrksiazki się powtórzy. I zajądą błędy. Ta sama sytuacja powtarza się z czytelnikiem, który jest tworzony na podobnej podstawie.

## 2) SPOSÓB DZIAŁANIA PROGRAMU

Program umożliwia interakcję z biblioteką, zarówno dla pracowników, jak i dla czytelników. Jego głównym celem jest umożliwienie wypożyczania książek, ich zwrotów, sprawdzania dostępności, zakupu nowych i domawiania pojedynczych egzemplarzy oraz rejestracji nowych czytelników. Użytkownik który uruchamia projekt ma do wyboru zalogowanie się na 2 konta

Jednym z nich jest sam czytelnik który może w bibliotece wykonać samodzielnie 3 czynności. Pierwszym wyborem jest znalezienie książek w bibliotece, chodzi tutaj m.in. o Wyświetlenie użytkownikowi regału oraz miejsca na konkretnej półce. Regały są z góry przydzielone konkretnemu rodzajowi, a miejsce na półce jest alfabetycznie poukładane tytułem książki i ich ilością. Wybierając te opcje mamy do wyboru znalezienie książki po tytule, autorze oraz rodzaju. Następnym zadaniem, które może wykonać czytelnik jest sprawdzenie swojego konta na którym jest pokazana ilość książek na stanie użytkownika. Użytkownik sprawdza swoje konto poprzez zalogowanie się numerem karty. Numer karty jest indywidualny dla każdego użytkownika i automatycznie generowany przy zakładaniu konta. W realnym świecie każda osoba korzystająca z biblioteki W Mińsku Mazowieckim,

ma indywidualną kartę która ma w sobie kod kreskowy I to za pomocą tego kodu Użytkownik jest w stanie wypożyczyć książkę. W moim w przypadku kod kreskowy zastępuje numer karty, który reprezentuje to samo zadanie. Przy wybraniu trzeciej opcji czytelnik może zarejestrować się w systemie bibliotecznym. Musi podać przy tym swoje imię, nazwisko oraz rok urodzenia (rok sprawdzany jest przez system, ma podany zakres od 1930 roku do teraźniejszego). Użytkownik który chce opuścić konto czytelnika wybiera opcję numer 4, która przechodzi do menu głównego.

Drugim kontem jest konto pracownika, lecz by do niego się dostać musimy się zalogować. Zalogować się możemy tylko o danych **imie = „natalia” oraz hasło=„zaq123”**. ( Moim założeniem jest jak najbardziej urzeczywistnienie programu, dlatego nie chciałam, aby użytkownik mógł sam utworzyć sobie konto pracownicze, tak jak może to zrobić dla konta czytelnika. W realnym świecie to informatycy, którzy odpowiadają za uprawnienia pracowników, systemy bazowe, bądź zarządzanie sieciami firmy, to ich zadaniem jest wprowadzenie pracownika do firmy pod względem systemu, uprawnień i przeszkolenia. Sama w funkcja, która by odpowiadała za dodawanie pracowników byłaby bardzo podobna do funkcji zakładania kont czytelników, dlatego też nie chciałam jeszcze bardziej poszerzać programu, gdy widziałam że w rzeczywistości to tak nie wygląda). Jak zalogujemy się na konto pracownika wyświetla się menu pracownika, w którym mamy do wyboru wiele zadań. Wszystkie te zadania są zapisane w oddzielnych funkcjach. Pierwszą funkcją jest wypożyczenie książki przez czytelnika. By to zrobić pracownik musi wprowadzić numer karty, którą dostaje od czytelnika. (By wypożyczyć książkę w rzeczywistości czytelnik musi pokazać kartę, która przedstawia jak wcześniej wspomniałam kod kreskowy). Następnie pracownik podaje numer książki, który w programie jest Identyfikatorem każdej książki (zazwyczaj taki ciąg cyfr znajduje się na pierwszej stronie w każdej książce). Dopiero po wprowadzeniu takich danych książka zostaje wypożyczona, czyli zapisana na konto czytelnika i oczywiście zostaje usunięta ze stanu dostępnych książek w bibliotece. Kolejną funkcją jest zwrot książki, który działa na tej samej zasadzie tylko nie wypożycza, a zwraca książkę i dopisuje ją do stanu dostępnych książek w bibliotece. Następnie przy wyborze numer 3, pracownik także może wyszukać książki w bibliotece, gdzie leżą i na jakiej półce. Następne 3 wybory, są do siebie bardzo podobne i polegają ostatecznie na tym samym, lecz różnią się wstępem. Zaczynając od wyboru numer 4, pracownik może sprawdzić stan książek w bibliotece. To znaczy, że weryfikuje ile książek brakuje do minimalnej ilości jaka musi się znaleźć. Wszystkie braki są wypisane na ekranie wraz z konkretnym identyfikatorem książki oraz ilością braków. Po tym wyświetleniu pracownik ma do wyboru czy chce z tego miejsca domówić brakujące egzemplarze, czy jednak chciał tylko sprawdzić brakującą ilość. Jednak jeśli wybierze opcję domówienia, to zostaje przekierowany do funkcji domówienia egzemplarzy, którą także może uruchomić wybierając z menu opcję numer 5. Gdy pracownik będzie domawiać egzemplarze, jest proszony o identyfikator książki, jaką chce domówić oraz ilość którą chce dołożyć do aktualnego stanu w bibliotece. W rzeczywistości nie dzieje się to tak szybko jak w programie, że wystarczy kliknąć Enter i książki są od razu na stanie w bibliotece. Chcąc zasymulować tą sytuację, że zamówienie jest pakowane w magazynie, czeka na kuriera, w drodze, czy czeka na odebranie przez zamawiającego. Użyłam ciekawego rozwiązania, a mianowicie procesu opóźnienia. Użytkownik w tym momencie przybiera postać różnych osób: pracownika, który wysyła zamówienie, magazynier kurier oraz z powrotem pracownik, który musi odebrać paczkę i zatwierdzić ją do systemu. Ważną rzeczą w programie jest by przy wyborze czy zamówienie zostało opłacone wybrać opcję tak. Ponieważ w rzeczywistości pracownik zakupując/domawiając książki nie płaci z własnych pieniędzy. Biblioteka ma podpisane ogólnie umowy z czytelniami oraz magazynami książek. A rozliczenia są roczne na

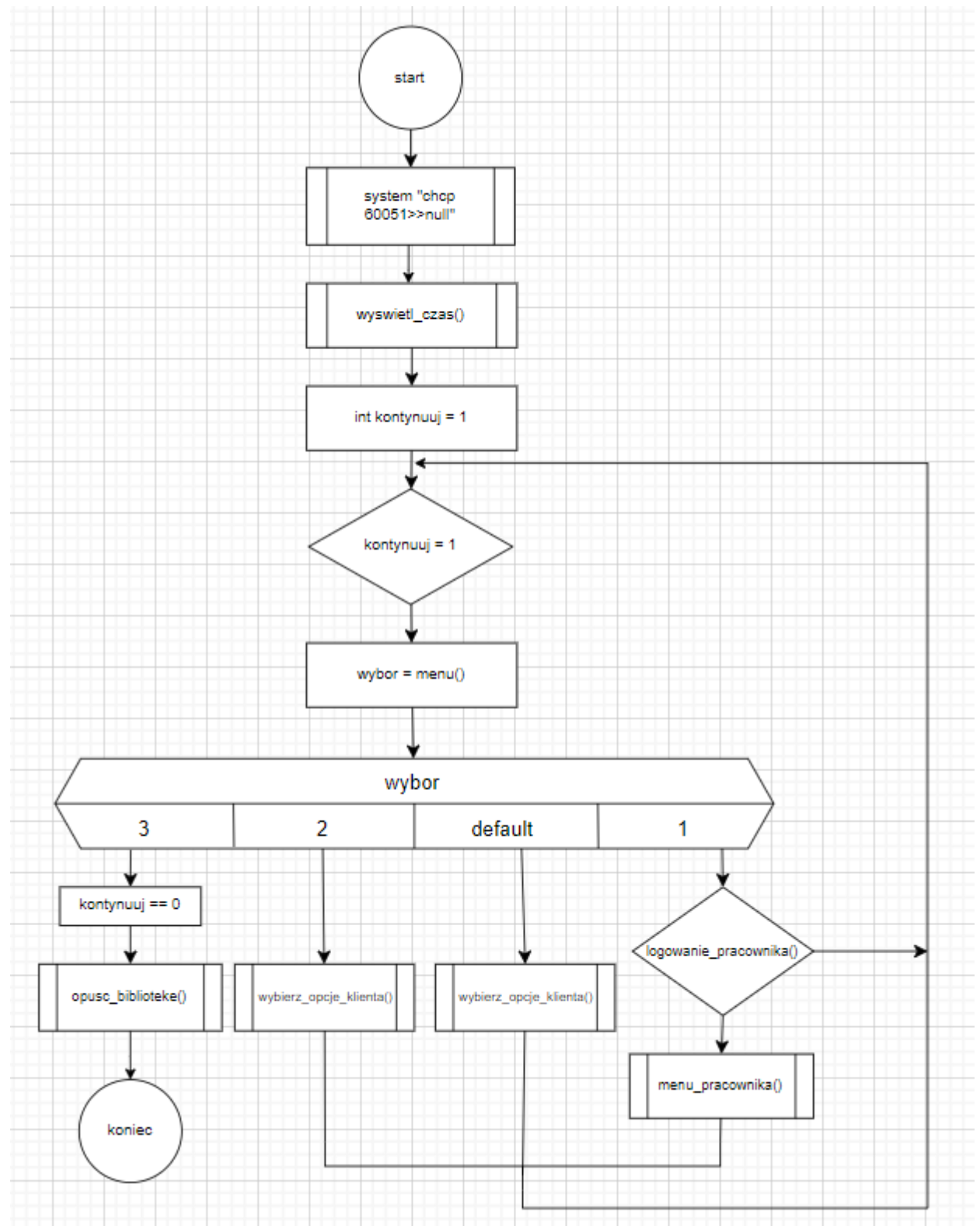
pokrycie kosztów. Więc z góry zakładamy, że zamówienie jest opłacone i wtedy zamówienie idzie do realizacji itd., zgodnie z kodem programu. A gdy wybierzemy, że nie jest opłacona funkcja się zakończy. Tą sytuację dobrze widać dopiero w czasie trwania programu, więc nie będę tu dużo o niej pisać tylko przy jej omówieniu, lecz jak widać już wyżej napisałam że funkcja ta wykorzystywana jest także przy zakupie książek nowy do biblioteki. Dzieje się tak gdy z menu wybiorę opcję numer 6, która odpowiedzialna jest za zakup książek. Pracownik podaje konkretne informacje o tytule, autorze, roku wydania, ilości minimalnej jaka musi być w bibliotece oraz ile aktualnie zamawia. Po tak wypełnionym formularzu zamówienie zostaje wysłane to znaczy, że uruchomiona jest funkcja którą opisałam powyżej, a mianowicie symulacja rzeczywistego zamówienia. Ostatnim wyborem dla pracownika jest opuszczenie konta, przez co cofa się do menu głównego gdzie także wybierając opcję nr 3 może całkowicie wyjść z programu.

Bardziej szczegółowe wyjaśnienie funkcji oraz plików, będzie w dalszej części pracy. Tutaj podałam ogólne założenia oraz sam schemat działania i możliwości jakie mamy do wyboru korzystając z programu.

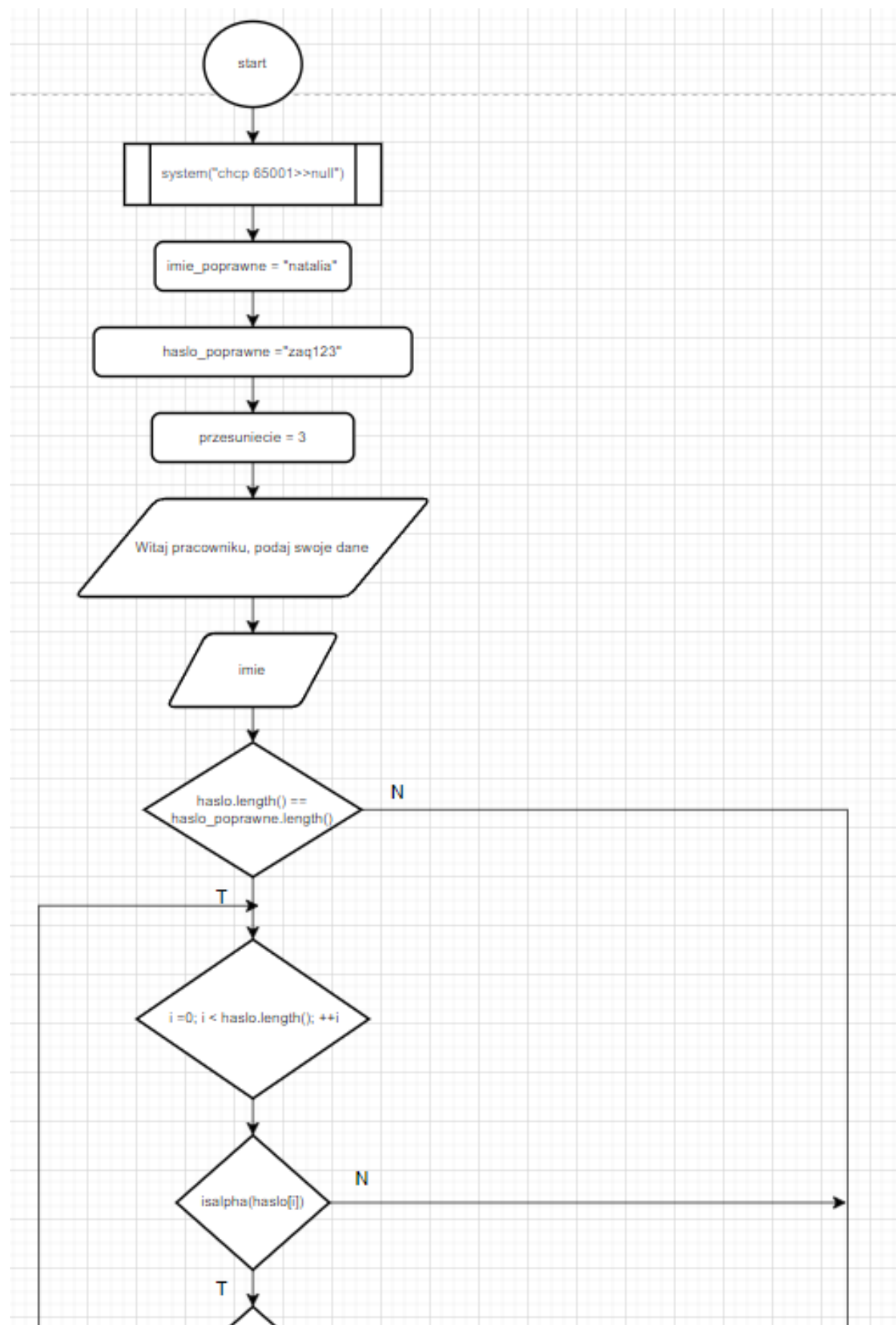
### **3) SCHEMAT BLOKOWY**

Do zrobienia schematów blokowych wykorzystuję program <https://app.diagrams.net>.

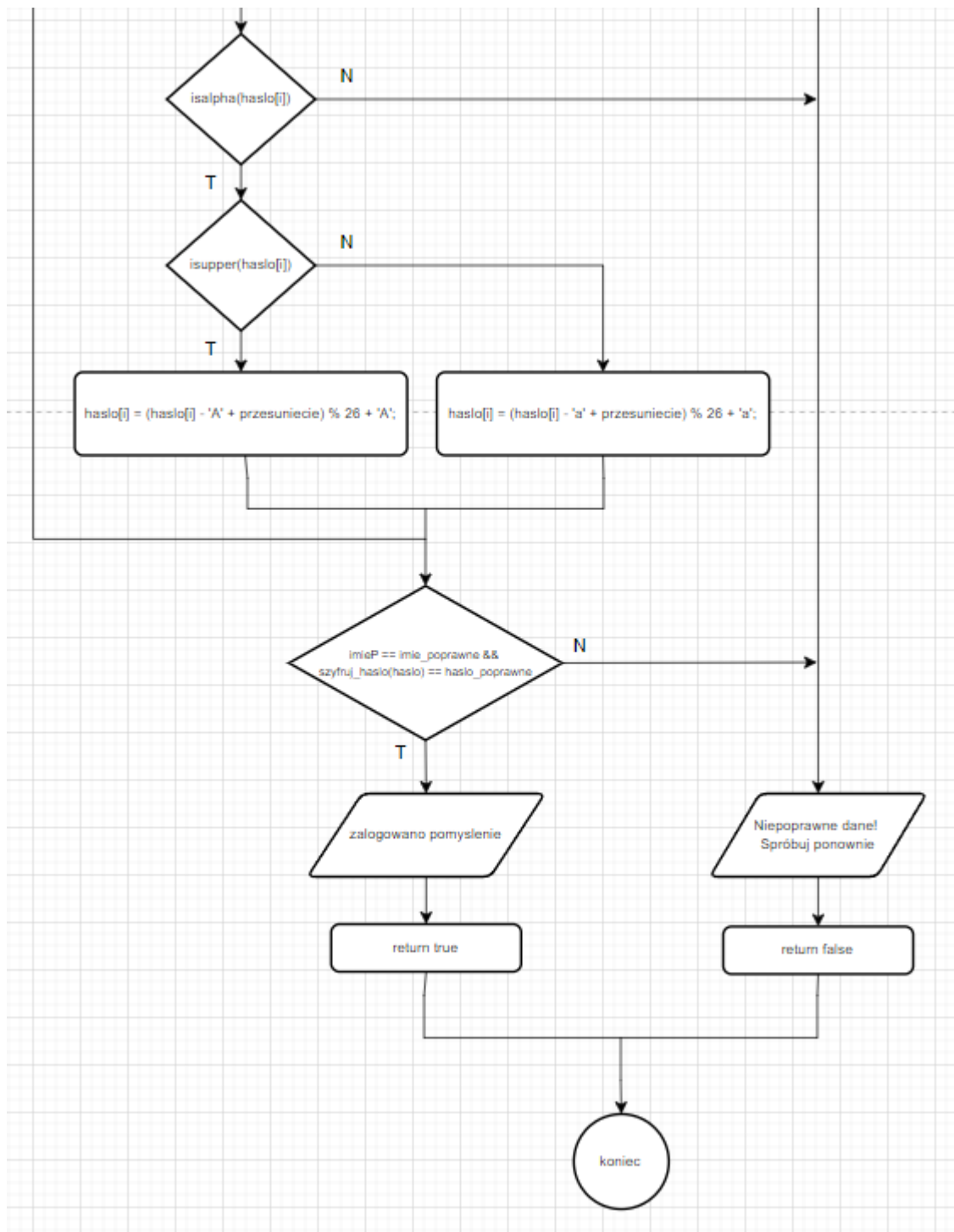
Schemat przedstawia funkcję w pliku główny `int main()`



Schemat przedstawia funkcje logowanie\_pracownika():







#### 4) WYJAŚNIENIE PLIKU GŁÓWNEGO: MAIN.CPP

Projekt rozpoczynamy od pliku głównego: main.cpp to do niego będziemy podpinąć pliki nagłówkowe zawierające funkcje z odpowiadającymi do nich kodami. W tym pliku jedynie w całym projekcie mamy funkcję główną 'int main()', która wywołuje zapisane funkcje.

Biblioteki jakich użyłam w tym pliku:

- ✓ <iostream> – do obsługi wejścia/wyjścia (wyświetlanie tekstów na ekranie);
- ✓ <ctime> – do pobierania czasu;
- ✓ <cstdlib> – do używania funkcji systemowych (polskie znaki);
- ✓ <string> – do pracy z łańcuchami znaków.
- ✓ <sstream> - Umożliwia pracę ze strumieniami na łańcuchach znaków

Pliki nagłówkowe:

- ✓ #include "rejestracjaK.h"
- ✓ #include "zadaniaNaKsiazkach.h"
- ✓ #include "bazaKsiazek.h"
- ✓ #include "znajdzKsiazke.h"
- ✓ #include "klient.h"
- ✓ #include "szyfrowanieHasla.h"

Funkcja główna programu: 'int main()'

To ona jest głową naszego programu. Uruchamiając po kolei zapisane w sobie funkcje opisane poniżej, będziemy poruszać się po całym programie. Po przedstawieniu daty oraz miejsca biblioteki. Wykonuje funkcję menu(), która jest łącznikiem do poruszania się po programie. To w niej wybieram na jakie konto chcę wejść, wybierając odpowiednią cyfrę. Gdy ma wybrane na jakie konto chce się zalogować, poprzez wprowadzenie odpowiedniego numeru, zostaje przekierowany do odpowiedniej funkcji. Jak widać menu\_pracownika() jest poprzedzone funkcją logowanie\_pracownika() i warunkiem if. Skutkuje to warunkiem że jeśli funkcja logowanie\_pracownika() nie zostanie spełniona, nie będziemy mogli przejść do konta pracownika. Tą funkcję opiszę już poniżej, ponieważ wykorzystuję w niej szyfrowanie hasła, więc przeznaczyłam na logowanie pracownika oddzielny plik. Ale możemy zauważyć, że w pliku głównym mamy przy wyborach czynności zrobione pętle by nasz program nie kończył się automatycznie po wykonaniu zadania a dał możliwość przejścia do kolejnego. Wykorzystuję pętlę while, która umożliwia mi ciągłą interakcję z użytkownikiem, aż do momentu kiedy użytkownik sam nie zdecyduje opuszczenia biblioteki, czyli wybór nr 3, w tym momencie moja pętla się zakończy i uruchomi ostatnią funkcję opusc\_biblioteke().

```

129
130 int main() {
131     system("chcp 65001>>null");
132     wyswietl_czas();
133
134     int wybor;
135     int kontynuuj = 1;
136     while (kontynuuj){
137         wybor = menu();
138         switch (wybor) {
139             case 1:
140                 if (logowanie_pracownika()){
141                     menu_pracownika();
142                 }
143                 break;
144             case 2:
145                 wybierz_opcje_klienta();
146                 break;
147             case 3:
148                 kontynuuj = 0;
149                 opusc_biblioteke();
150                 break;
151             default:
152                 blad();
153                 break;
154         }
155     }

```

Funkcje pomocnicze:

- wyswietlenie\_czasu() – wyświetla aktualną datę w formacie dzień:miesiąc:rok informując o tym użytkownika na samym początku programu oraz witając się z nim i przedstawiając bibliotekę. Używa standardowej funkcji time() do pobrania aktualnego czasu w sekundach, a następnie konwertuje go na format lokalny za pomocą funkcji localtime(). Data jest formatowana za pomocą strftime(), a wynik jest wyświetlany na ekranie poprzez wywołanie wyswietl.

```

16 void wyswietl_czas() {
17     time_t timestamp = time(NULL);
18     struct tm datetime = *localtime(&timestamp);
19     char wyswietl[50];
20     strftime(wyswietl, 50, "%d:%m:%Y", &datetime);
21     cout << "Witam, dziś jest: " << wyswietl << endl;
22     cout << "Miło cię widzieć w naszej bibliotece miejskiej w Mińsku Mazowieckim.";
23 }

```

- menu() – wyświetla główne menu programu, oferując użytkownikowi wybór pomiędzy kontem pracownika, kontem czytelnika lub wyjściem z programu. Ta funkcja jedynie wyświetla nam menu, a sam wybór konta i reszta powiązanego kodu znajduje się w funkcji głównej.

```

25 int menu() {
26     int wybor;
27     cout << "\n Wybierz cyfrę, która poprowadzi cię na odpowiednie konto: " << endl;
28     cout << "1. Pracownik" << endl;
29     cout << "2. Czytelnik" << endl;
30     cout << "3. Opuść bibliotekę" << endl;
31     cout << "Podaj wybór: "; cin >> wybor;
32     return wybor;
33 }

```

Wynik powyższych funkcji:

```

C:\Users\natal\Desktop\popr:
Witam, dziś jest: 30:01:2025
Miło cię widzieć w naszej bibliotece miejskiej w Mińsku Mazowieckim.
Wybierz cyfrę, która poprowadzi cię na odpowiednie konto:
1. Pracownik
2. Czytelnik
3. Opuść bibliotekę
Podaj wybór: |

```

Funkcja związane z kontem pracownika:

- menu\_pracownika() – funkcja umożliwiająca pracownikowi biblioteki wybór jednej z opcji , które chce wykonać w danym momencie. Pod każdy numer jaki użytkownik wybierze jest odwołanie się do funkcji, która jest za to odpowiedzialna. Funkcje znajdują się w innych plikach nagłówkowych, które podawałam powyżej. Pracownik może wykonać jedną z czynności poprzez wybór odpowiedniej opcji z menu. Dzięki wykorzystaniu pętli while, mamy możliwość do ponownego wyboru, chyba że chcemy opuścić to konto to funkcja się nie powtórzy. Pracownik może wywołać funkcje takie jak: wypożycz\_książke(), zwroc\_książke(), stan\_książek(), zakup\_książki(), domow\_książke(), znajd\_książke() oraz zadanie opuszczenia konta pracownika., które nie wykorzystuje funkcji tylko cofa się do menu().

```

32 void menu_pracownika() {
33     int pracownik;
34     int kontynuuj = 1;
35
36     while (kontynuuj) {
37         cout << endl;
38         cout << "Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać." << endl;
39         cout << "1. Wypożyczyć książkę czytelnikowi" << endl;
40         cout << "2. Zwrot książki" << endl;
41         cout << "3. Znaleźć miejsce książki w bibliotece" << endl;
42         cout << "4. Sprawdzić brakujące egzemplarze książek" << endl;
43         cout << "5. Dokup egzemplarze książek" << endl;
44         cout << "6. Zamów nowe książki do biblioteki" << endl;
45         cout << "7. Wylogować się z konta pracownika" << endl;
46         cout << "Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: "; cin >> pracownik;
47         cout << endl;
48
49         switch (pracownik) {
50             case 1:
51                 wypożycz_książke();
52                 break;
53             case 2:
54                 zwroc_książke();
55                 break;
56             case 3:
57                 znajdź_książke();
58                 break;
59             case 4:
60                 stan_książek();
61                 break;
62             case 5: {
63                 Biblioteka biblioteka;
64                 domów_książke(biblioteka);
65                 break;
66             }
67             case 6: {
68                 Biblioteka biblioteka;
69                 biblioteka.zakup_książki();
70                 break;
71             }
72             case 7:
73                 kontynuuj = 0;
74                 cout << "Dziękuję za dzisiejszą pracę" << endl;
75                 menu();
76                 break;
77             default:
78                 cout << "Niepoprawny wybór!" << endl;
79             }
80         }
81     }
82 }

```

Wynik wyboru funkcji menu\_pracownika():

```

C:\Users\natal\Desktop\baza_ x + v
Miło cię widzieć w naszej bibliotece miejskiej w Mińsku Mazowieckim.
Wybierz cyfrę, która poprowadzi cię na odpowiednie konto:
1. Pracownik
2. Czytelnik
3. Opuść bibliotekę
Podaj wybór: 1
Witaj pracowniku, podaj swoje dane
Imię: nataliaa
Podaj hasło: *****
Niepoprawne dane! Spróbuj ponownie

Wybierz cyfrę, która poprowadzi cię na odpowiednie konto:
1. Pracownik
2. Czytelnik
3. Opuść bibliotekę
Podaj wybór: 1
Witaj pracowniku, podaj swoje dane
Imię: natalia
Podaj hasło: *****
Zalogowano pomyślnie!

Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać.
1. Wypożyczyć książkę czytelnikowi
2. Zwrot książki
3. Znaleźć miejsce książki w bibliotece
4. Sprawdzić brakujące egzemplarze książek
5. Dokup egzemplarze książek
6. Zamów nowe książki do biblioteki
7. Wylogować się z konta pracownika
Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: |

```

Funkcja związana z kontem czytelnika:

Wybierz\_opcje\_klienta() – ponownie ustawiam na samym początku pętli while, która umożliwia mi powrót do menu pracownika. Chyba że użytkownik będzie chciał cofnąć się do menu głównego wtedy zmieniam wartość kontynuuj na 0 i program sam cofa się do menu(). Gdy wybiorę inną cyfrę to analogicznie do wyświetlonej treści menu zostaje odwołana do takiej funkcji, która jest przypisana do wybranej cyfry. Wszystkie te funkcje z wyjątkiem menu() są zapisane w innych plikach nagłówkowych, by zachować przejrzystość kodu. Funkcje jakie wykorzystywane są po wyborze tego konta: znajdź\_książkę(), sprawdź\_konto\_klienta(), rejestracja\_klienta(), menu().

```
84
85 void wybierz_opcje_klienta() {
86     int klient;
87     int kontynuuj = 1;
88
89     while (kontynuuj) {
90         cout << endl;
91         cout << "Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj: " << endl;
92         cout << "1. Znaleźć położenie książki w bibliotece" << endl;
93         cout << "2. Sprawdzić ilość książek na swoim koncie" << endl;
94         cout << "3. Zarejestrować się" << endl;
95         cout << "4. Wrócić do menu głównego" << endl;
96         cout << "Jeśli dokonales wyboru, wpisz cyferkę przypisaną czynności: "; cin >> klient;
97         cout << endl;
98
99         switch (klient) {
100             case 1:
101                 znajdź_książkę();
102                 break;
103             case 2:
104                 sprawdź_konto_klienta();
105                 break;
106             case 3:
107                 rejestracja_klienta();
108                 break;
109             case 4:
110                 kontynuuj = 0;
111                 break;
112             default:
113                 cout << "Niepoprawny wybór!" << endl;
114         }
115     }
116 }
117
```

Wynik wyboru funkcji wybierz\_opcje\_klienta():

```
C:\Users\natal\Desktop\popr: X + v
Witam, dziś jest: 30:01:2025
Miło cię widzieć w naszej bibliotece miejskiej w Mińsku Mazowieckim.
Wybierz cyfrę, która poprowadzi cię na odpowiednie konto:
1. Pracownik
2. Czytelnik
3. Opuść bibliotekę
Podaj wybór: 2

Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj:
1. Znaleźć położenie książki w bibliotece
2. Sprawdzić ilość książek na swoim koncie
3. Zarejestrować się
4. Wrócić do menu głównego
Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 5

Niepoprawny wybór!

Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj:
1. Znaleźć położenie książki w bibliotece
2. Sprawdzić ilość książek na swoim koncie
3. Zarejestrować się
4. Wrócić do menu głównego
Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 4

Wybierz cyfrę, która poprowadzi cię na odpowiednie konto:
1. Pracownik
2. Czytelnik
3. Opuść bibliotekę
Podaj wybór: |
```

## 5) WYJAŚNIENIE PLIKI NAGŁÓWKOWEGO SZYFROWANIEHASLA.H

Przedstawiony kod jest częścią programu, który implementuje funkcję logowania dla pracownika przy użyciu hasła szyfrowanego. Program zawiera funkcję do wczytywania hasła od użytkownika oraz funkcję realizującą proces logowania i szyfrowania. Hasło jest szyfrowane metodą szyfru Cezara, a dane logowania są weryfikowane przez porównanie z wcześniej określonymi wartościami.

Biblioteki z jakich korzystam:

- #include <iostream>
- #include <string>
- #include <conio.h>

- Funkcja wczytaj\_haslo() - służy do wczytywania hasła od użytkownika. Używa funkcji \_getch(), która pobiera pojedynczy znak bez wyświetlania go na ekranie. Wprowadzone znaki są zastępowane przez symbol \*, co ma na celu zwiększenie

bezpieczeństwa przy wprowadzaniu hasła. Obsługuje także przypadek, w którym użytkownik chce usunąć ostatnio wprowadzony znak, wykorzystując kod 8 (Backspace). Hasło jest przechowywane w zmiennej `haslo`, którą funkcja zwraca po zakończeniu wprowadzania.

Zastosowałam tutaj dwie funkcje, które są warte uwagi:

- ✓ `empty()` - która sprawdza, czy ciąg znaków jest pusty. Zwraca wartość typu `bool`: `true`, jeśli ciąg jest pusty (nie zawiera żadnych znaków), `false`, jeśli ciąg zawiera co najmniej jeden znak.
- ✓ `Pop_back()` - usuwa ostatni znak z ciągu znaków. Jeśli ciąg nie jest pusty, metoda ta zmienia zawartość stringa, usuwając ostatni znak. Nie zwraca żadnej wartości.
- ✓ `Push_back()` - zamienia mi wprowadzane litery na \*

```
9
10 string wczytaj_haslo() {
11     string haslo = "";
12     char ch;
13
14     cout << "Podaj haslo: ";
15     while ((ch = _getch()) != '\r') {
16         if (ch == 8) {
17             if (!haslo.empty()) {
18                 haslo.pop_back();
19                 cout << "\b \b";
20             }
21         } else {
22             haslo.push_back(ch);
23             cout << "*";
24         }
25     }
26     cout << endl;
27     return haslo;
28 }
```

#### Funkcja `logowanie_pracownika()`

Funkcja `logowanie_pracownika()` – Realizuję proces logowania pracownika, który składa się z dwóch kroków: wprowadzenia poprawnego imienia i poprawnego hasła. Program ustawia stałe wartości dla poprawnego imienia (`imie_poprawne = "natalia"`) oraz dla szyfrowanego hasła (`haslo_poprawne = "zaq123"`). Użytkownik wprowadza swoje imię, które jest porównywane z poprawnym imieniem. Następnie wprowadza hasło, które jest szyfrowane i porównywane z wcześniej zaszyfrowanym hasłem. Hasło przyjmowane jest w postaci ciągu znaków (`string`) i zwraca zaszyfrowaną wersję tego hasła. Szyfrowanie odbywa się przy użyciu szyfru Cezara, w którym każde wystąpienie litery w hasle jest przesunięte o stałą liczbę pozycji, w tym przypadku 3. Jeżeli znak jest literą dużą, to przesunięcie dotyczy tylko tych liter. Jeżeli znak jest literą małą, przesunięcie odbywa się tylko w zakresie małych liter. Przesunięcie 3 oznacza, że każda litera w hasle zostanie zamieniona na literę znajdującą się o 3 miejsca dalej w alfabecie. Inne znaki, takie jak cyfry czy znaki interpunkcyjne, nie są zmieniane. Jeśli zarówno imię, jak i hasło są poprawne, wyświetlana jest wiadomość "Zalogowano pomyślnie!". W przeciwnym przypadku wyświetlany jest komunikat o błędnych danych oraz masz możliwość ponownego zalogowania.



```

30 bool logowanie_pracownika() {
31     system("chcp 65001>>null");
32     string imieP;
33     string haslo;
34
35     const string imie_poprawne = "natalia";
36     const string haslo_poprawne = "zaql23";
37     int przesuniecie = 3;
38
39     cout << "Witaj pracowniku, podaj swoje dane" << endl;
40     cout << "Imię: "; cin >> imieP;
41
42     if(haslo.length() == haslo_poprawne.length()){
43         for (int i =0; i < haslo.length(); ++i) {
44             if (isalpha(haslo[i])) {
45                 if (isupper(haslo[i])) {
46                     haslo[i] = (haslo[i] - 'A' + przesuniecie) % 26 + 'A';
47                 } else {
48                     haslo[i] = (haslo[i] - 'a' + przesuniecie) % 26 + 'a';
49                 }
50             }
51         }
52         return false;
53     }
54     haslo = wczytaj_haslo();
55     if (imieP == imie_poprawne && haslo == haslo_poprawne) {
56         cout << "Zalogowano pomyślnie!" << endl;
57         return true;
58     } else {
59         cout << "Niepoprawne dane! Spróbuj ponownie" << endl;
60         return false;
61     }
62 }
63

```

Rezultat tego pliku:

```
C:\Users\natal\Desktop\baza x + v
Miło cię widzieć w naszej bibliotece miejskiej w Mińsku Mazowieckim.
Wybierz cyfrę, która poprowadzi cię na odpowiednie konto:
1. Pracownik
2. Czytelnik
3. Opuść bibliotekę
Podaj wybór: 1
Witaj pracowniku, podaj swoje dane
Imię: nataliaa
Podaj hasło: *****
Niepoprawne dane! Spróbuj ponownie

Wybierz cyfrę, która poprowadzi cię na odpowiednie konto:
1. Pracownik
2. Czytelnik
3. Opuść bibliotekę
Podaj wybór: 1
Witaj pracowniku, podaj swoje dane
Imię: natalia
Podaj hasło: *****
Zalogowano pomyślnie!

Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać.
1. Wypożyczyć książkę czytelnikowi
2. Zwrot książki
3. Znaleźć miejsce książki w bibliotece
4. Sprawdzić brakujące egzemplarze książek
5. Dokup egzemplarze książek
6. Zamów nowe książki do biblioteki
7. Wylogować się z konta pracownika
Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: |
```

## 6) WYJAŚNIENIA DO PLIKU BAZAKSIAZEK.H

Jest to główny plik związany z pracą na książkach. Program symuluje proces zakupu i rejestracji książek w bibliotece. Użytkownik wprowadza dane o książce tytuł, autor, wybiera jej rodzaj, ustala stan magazynowy, a następnie książka przechodzi przez proces zamówienia, dostawy i rozłożenia na półkach w bibliotece. Cały proces zawiera symulację opóźnień na każdym etapie, które są realizowane poprzez wątki, a także obsługę błędów (np. błędne odpowiedzi od użytkownika). Aby dostać się tej funkcji i wykonać na niej prace muszą przejść do menu pracownika, zalogować się oraz wybrać opcję 6, która przedstawia zadanie: „zamów nowe książki do biblioteki”.

Używam także obiektywności, w której atrybutami prywatnymi są: tytuł, autor, rodzaj, rok wydania, ilość minimalna, stan, numer książki.

Biblioteki jakich używam:

- `#include <iostream>`
- `#include <cstdlib>`
- `#include <chrono>`
- `#include <thread>` - wątki, które są jednostkami wykonywania w programie. Wątki pozwalają na równoczesne wykonywanie różnych części programu
- `#include <fstream>`

### Funkcja zakup\_książki()

Funkcja zakup\_książki() – metoda publiczna, odpowiedzialna za zbieranie informacji o książce, którą pracownik chce dodać do biblioteki. Pracownik podaje tytuł, autora, rok wydania, minimalną liczbę egzemplarzy, jakie mają być dostępne w bibliotece, oraz liczbę zamawianych książek. Rodzaj nie jest wpisywany przez użytkownika tylko odesłani jesteśmy do funkcji wybierz\_rodzaj(). Następnie funkcja ustala numer książki, ten proces także ma ustaloną funkcję, która pobiera dane nrKsiążki. Gdy przejdziemy przez wprowadzenie wszystkich tych danych funkcja przekazuje dane do funkcji zloz\_zamowienie(), która kontynuuje proces.

```
17 void zakup_książki() {
18     system("chcp 65001>>null");
19
20     cout << "Podaj tytuł książki: "; cin >> tytuł;
21     cout << "Podaj autora książki: "; cin >> autor;
22     cout << "Podaj rok wydania książki: "; cin >> rok_wydania;
23     rodzaj = wybierz_rodzaj();
24     cout << "Podaj minimalną ilość jaka musi być na stanie w bibliotece: "; cin >> ilosc_min;
25     cout << "Podaj ile kupujesz egzemplarzy książki: "; cin >> stan;
26     nrKsiążki = odczytaj_ostatni_nr_książki() + 1;
27
28     zloz_zamowienie();
29 }
```

Wynik powyższej funkcji:

```
Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać.
1. Wypożyczyć książkę czytelnikowi
2. Zwrot książki
3. Znaleźć miejsce książki w bibliotece
4. Sprawdzić brakujące egzemplarze książek
5. Dokup egzemplarze książek
6. Zamów nowe książki do biblioteki
7. Wylogować się z konta pracownika
Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 6

Podaj tytuł książki: dsd
Podaj autora książki: dsd
Podaj rok wydania książki: 2002
Wybierz rodzaj książki:
1. Bajka
2. Powieść
3. Kryminal
4. Fantastyka
5. Kulinarne
Wybierz odpowiedni numer: 3
Podaj minimalną ilość jaka musi być na stanie w bibliotece: 5
Podaj ile kupujesz egzemplarzy książki: 36
Zamówienie zostało wysłane.
```

- odczytaj\_ostatni\_nr\_książki(): metoda prywatna, odczytuje numer ostatniej książki z pliku "numer\_książki.txt", aby nadać nowej książce numer kolejny. Jeśli plik nie istnieje, zaczyna numerację od 901.
- zapisz\_numer\_książki(int nrKsiążki): zapisuje numer książki do pliku "numer\_książki.txt". Przy każdorazowym zakupie numer książki jest inkrementowany i zapisany w pliku, zapewniając unikalność numerów. Zapisanie (int nrKsiążki) oznacza, że funkcja oczekuje przyjęcia numeru książki jako argumentu, aby mogła wykonać operację w tym przypadku zapisanie go do pliku.

- `wybierz_rodzaj()`: metoda prywatna, umożliwia użytkownikowi wybranie rodzaju książki spośród dostępnych opcji (bajka, powieść, kryminał, fantastyka, kulinarna). Zwraca nazwę wybranego rodzaju. Wyboru dokonujemy poprzez wybranie cyfry, która odpowiada danemu rodzajowi.

```
64 string wybierz_rodzaj() {
65     int wybor_rodzaju;
66     string rodzaj;
67
68     cout << "Wybierz rodzaj książki:" << endl;
69     cout << "1. Bajka" << endl;
70     cout << "2. Powieść" << endl;
71     cout << "3. Kryminał" << endl;
72     cout << "4. Fantastyka" << endl;
73     cout << "5. Kulinarna" << endl;
74     cout << "Wybierz odpowiedni numer: "; cin >> wybor_rodzaju;
75
76     switch(wybor_rodzaju) {
77         case 1: rodzaj = "bajka"; break;
78         case 2: rodzaj = "powiesc"; break;
79         case 3: rodzaj = "kryminal"; break;
80         case 4: rodzaj = "fantastyka"; break;
81         case 5: rodzaj = "kulinarne"; break;
82         default: cout << "Niepoprawny wybór!"; break;
83     }
84     return rodzaj;
85 }
86
87 void zloz_zamowienie()
```

Teraz przejdźmy do opisanie funkcji, które odpowiedzialne są za złożenie zamówienia. Te funkcje wykorzystywane są nie tylko w funkcji `zakup_książki()`, ale także `domow_książke()`. Chciałam tutaj pokazać jak najbardziej urzeczywistnioną sytuację związaną z zamówieniem paczki (dla nas jest to książka). Gdy przejdziemy przez wypełnienie informacji w funkcji `zakup_książki()`/`domow_książke()` to na samym końcu wywołuje funkcję `zloz_zamowienie()`. Odpowiada ona za wyświetlanie krótkich informacji i daje możliwość użytkownikowi wybór między „tak” a „nie”. Użytkownik w tym momencie przybiera postać równych osób. Trzeba pamiętać o zaznaczeniu tak przy wyborze czy zamówienie zostało opłacone. Z góry zakładamy, że tak i wtedy zamówienie idzie do realizacji itd. A gdy wybierzemy, że nie jest opłacone funkcja się zakończy. We wstępie opisałam swoje założenia do pracy.

W tej części kodu zastosowałam rozwiązanie, które symuluje dostarczenie paczki, a mianowicie proces opóźnienia. Realizowany za pomocą funkcji `this_thread::sleep_for()`, która wprowadza pauzę w działaniu programu na określony czas. Poleceniem **`this_thread::sleep_for(chrono::seconds(5))`** - wprowadzam pauzę trwającą 5 sekund, a przez wykorzystanie pętli `while` uruchamiam ją 2 razy

#### Funkcja `zloz_zamowienie()`

Funkcja `zloz_zamowienie()`: wyświetla informację o złożeniu zamówienia i pyta użytkownika, czy zamówienie zostało opłacone. Jeśli tak, uruchamia wątek realizacji zamówienia (`opoznienie1()`), jeśli nie, zamówienie zostaje anulowane. Ten wątek symuluje opóźnienie w procesie realizacji zamówienia, drukując komunikaty „Kompletowanie zamówienia...” co 5 sekund przez 2 powtórzenia. Po zakończeniu symulacji uruchamia

funkcję kurier(). Użyte zostało także czyszczenie wyświetlacza by jeszcze bardziej odczuć zmianę pracownika na magazyniera itd.

```
32
33 void zloz_zamowienie() {
34     cout << "Zamówienie zostało wysłane.\n";
35     this_thread::sleep_for(chrono::seconds(5));
36     system("cls");
37
38     char decyzja;
39     cout << "Czy zamówienie zostało opłacone? (t/n): "; cin >> decyzja;
40     if (decyzja == 't' || decyzja == 'T') {
41         cout << "Zamówienie zostało przyjęte. Rozpoczynamy realizację zamówienia...\n";
42         thread t(&Biblioteka::opoznienie1, this); // Uruchamiamy wątek realizacji
43         t.join(); // Czekamy na zakończenie wątku
44     } else {
45         cout << "Zamówienie nie zostało przyjęte. Paczka zostaje anulowana.\n";
46         return;
47     }
48 }
49
```

```
108 void opoznienie1() {
109     int licznik = 0;
110     while (licznik < 2) {
111         cout << "Kompletowanie zamówienia... (" << (licznik + 1) << "/2)\n";
112         this_thread::sleep_for(chrono::seconds(5));
113         licznik++;
114     }
115     cout << "Paczka jest gotowa do wysyłki!"<<endl;
116     cout << endl;
117     kurier();
118 }
119
```

- Funkcja kurier() – funkcja działa na tej samej zasadzie co powyższa, tylko zmienia komunikaty, przedstawiając czynności jakie wykonuje kurier. Wykorzystuje także proces symulacji opóźnienia uruchamiając funkcję opoznienie2(). Można zauważyć, że został użyty warunek, który daje mi możliwość tylko 3 razy wybrania opcji „nie”, jeśli tak się stanie paczka zostaje anulowana i wracamy do menu\_pracownika().

```
120 void kurier() {
121     char decyzja;
122     int prob = 0;
123     while (prob < 3) {
124         cout << "Czy paczka została odebrana przez kuriera? (t/n): ";
125         cin >> decyzja;
126
127         if (decyzja == 't' || decyzja == 'T') {
128             cout << "Paczka została odebrana przez kuriera. Kontroluj trasę paczki...\n";
129             opoznienie2();
130             break;
131         } else if (decyzja == 'n' || decyzja == 'N') {
132             prob++;
133             if (prob >= 3) {
134                 cout << "Przekroczono maksymalną liczbę prób. Paczka zostaje anulowana.\n";
135                 break;
136             }
137             cout << "Paczka nie została odebrana. Czekamy na kuriera...\n";
138             this_thread::sleep_for(chrono::seconds(5));
139         } else {
140             cout << "Nieprawidłowa odpowiedź. Proszę wpisać 't' lub 'n'. \n";
141         }
142     }
143 }
144
```

```

144
145 void opoznienie2() {
146     int licznik = 0;
147     while (licznik < 2) {
148         cout << "Zamówienie w drodze... (" << (licznik + 1) << "/2)\n";
149         this_thread::sleep_for(chrono::seconds(5));
150         licznik++;
151     }
152     cout << "Paczka dostarczona do miejsca docelowego!" << endl;
153     cout << endl;
154     dostarczenie();
155 }
156

```

- Funkcja dostarczenie() – pyta użytkownika, czy paczka została odebrana przez zamawiającego. Po otrzymaniu pozytywnej odpowiedzi uruchamia kolejne opóźnienie i wywołuje funkcję opoznienie3(). Jeśli paczka nie została odebrana, użytkownik ma 3 próby, po tych próbach zamówienie zostaje anulowane.

```

157 void dostarczenie() {
158     char decyzja;
159     int prob = 0;
160     while (prob < 3) {
161         cout << "Czy paczka została odebrana przez zamawiającego? (t/n): ";
162         cin >> decyzja;
163
164         if (decyzja == 't' || decyzja == 'T') {
165             cout << "Paczka została odebrana. Czekamy na zapisanie książek do bazy biblioteki.\n";
166             opoznienie3();
167             break;
168         } else if (decyzja == 'n' || decyzja == 'N') {
169             prob++;
170             if (prob >= 3) {
171                 cout << "Przekroczono maksymalną liczbę prób. Paczka zostaje anulowana.\n";
172                 break;
173             }
174             cout << "Paczka nie została odebrana. Czekamy na odebranie zamówienia...\n";
175             this_thread::sleep_for(chrono::seconds(5));
176         } else {
177             cout << "Nieprawidłowa odpowiedź. Proszę wpisać 't' lub 'n'. \n";
178         }
179     }
180 }
181

```

```

181
182 void opoznienie3() {
183     int licznik = 0;
184     while (licznik < 2) {
185         cout << "Zatwierdzanie egzemplarzy.... (" << (licznik + 1) << "/2)\n";
186         this_thread::sleep_for(chrono::seconds(5));
187         licznik++;
188     }
189     cout << "Książki są w bazie bibliotecznej!" << endl;
190     cout << endl;
191     rozlozenie();
192 }
193

```

- Funkcja rozlozenie() – użytkownik dostaje pytanie, czy książki zostały rozłożone na półkach bibliotecznych. Jeśli tak, książki zostają zapisane do bazy, a program kończy się. Jeśli nie, użytkownik ma 3 próby na udzielenie odpowiedzi „nie”. Po wyczerpaniu prób program zostaje zakończony. A jeśli odpowiemy „tak”, wyświetla nam się komunikat o rozłożeniu książek na półkę oraz zostajemy przekierowani do funkcji zapisz\_ksiazke().



```

194 void rozlozenie() {
195     char decyzja;
196     int prob = 0;
197     while (prob < 3) {
198         cout << "Czy książki zostały rozłożone na półki? (t/n): ";
199         cin >> decyzja;
200
201         if (decyzja == 't' || decyzja == 'T') {
202             cout << "Książki znajdują się na odpowiednich półkach.\n";
203             this_thread::sleep_for(chrono::seconds(5));
204             system("cls");
205             zapisz_książke();
206             return;
207         } else if (decyzja == 'n' || decyzja == 'N') {
208             prob++;
209             if (prob >= 3) {
210                 cout << "Przekroczono maksymalną liczbę prób. Program zostaje zakończony.\n";
211                 break;
212             }
213             cout << "Książki nie zostały rozłożone. Czekamy na dalsze informacje...\n";
214             this_thread::sleep_for(chrono::seconds(5));
215         } else {
216             cout << "Nieprawidłowa odpowiedź. Proszę wpisać 't' lub 'n'. \n";
217         }
218     }
219 }
220 };

```

- Funkcja zapisz\_książke() – dopiero ta funkcja umożliwia zapisanie danych o książce (tytuł, autor, rok wydania, rodzaj, ilość na stanie i numer książki) do pliku "ksiazki.txt". Po zapisaniu danych aktualizuje numer książki w pliku "numer\_ksiazki.txt". Dane o książce zapisane są w sposób przedstawiony poniżej. Jest to funkcja tak naprawdę odpowiedzialna ze zapisanie zamówionej książki, bądź dopisanie dokupionych egzemplarzy tylko wtedy gdy zamówienie zostanie dostarczone.

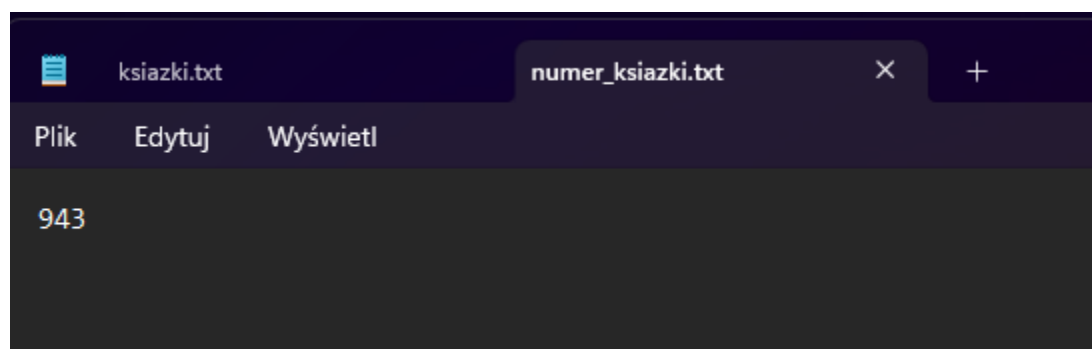
```

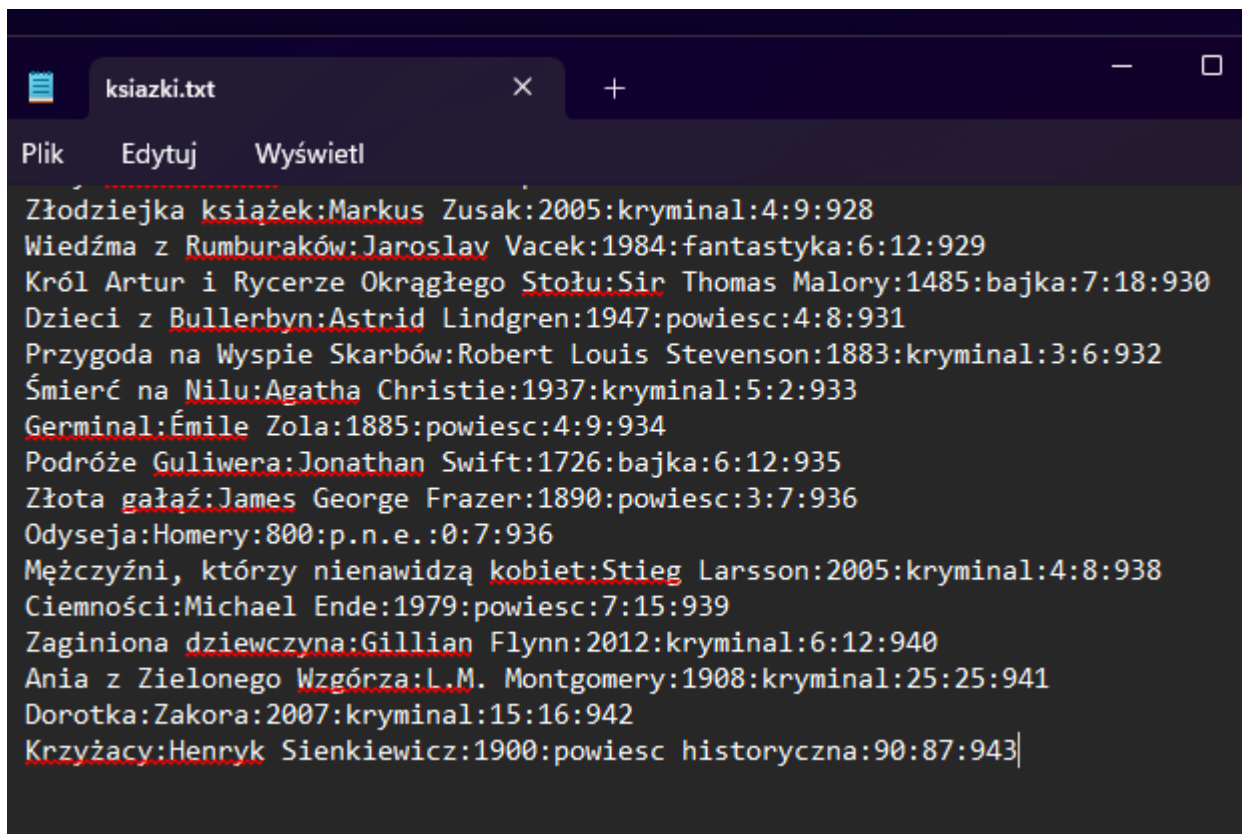
72 void zapisz_książke() {
73     ofstream plik("ksiazki.txt", ios::app);
74     if (plik.is_open()) {
75         plik << tytuł << ":" << autor << ":" << rok_wydania << ":" << rodzaj << ":" << ilosc_min << ":" << stan << ":" << nrKsiążki << endl;
76         cout << "Zamówiono do biblioteki książkę " << tytuł << endl;
77     } else {
78         cout << "Błąd przy zapisywaniu danych o książce!" << endl;
79     }
80     zapisz_numer_ksiazki(nrKsiążki);
81 }
82 }
83

```

Wyniki:

Tak wyglądają pliki przed zamówieniem:





Krok 1 przechodzę do zamówienie

```
Podaj hasło: *****
Zalogowano pomyślnie!

Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać.
1. Wypożyczyć książkę czytelnikowi
2. Zwrot książki
3. Znaleźć miejsce książki w bibliotece
4. Sprawdzić brakujące egzemplarze książek
5. Dokup egzemplarze książek
6. Zamów nowe książki do biblioteki
7. Wylogować się z konta pracownika
Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 6

Podaj tytuł książki: Pinokio
Podaj autora książki: Dobrzyński
Podaj rok wydania książki: 2003
Wybierz rodzaj książki:
1. Bajka
2. Powieść
3. Kryminal
4. Fantastyka
5. Kulinarna
Wybierz odpowiedni numer: 1
Podaj minimalną ilość jaka musi być na stanie w bibliotece: 20
Podaj ile kupujesz egzemplarzy książki: 15
Zamówienie zostało wysłane.
```



```
C:\Users\natal\Desktop\popr X + v
Czy zamówienie zostało opłacone? (t/n): t
Zamówienie zostało przyjęte. Rozpoczynamy realizację zamówienia...
Kompletowanie zamówienia... (1/2)
Kompletowanie zamówienia... (2/2)
Paczka jest gotowa do wysyłki!

Czy paczka została odebrana przez kuriera? (t/n): n
Paczka nie została odebrana. Czekamy na kuriera...
Czy paczka została odebrana przez kuriera? (t/n): t
Paczka została odebrana przez kuriera. Kontroluj trasę paczki...
Zamówienie w drodze... (1/2)
Zamówienie w drodze... (2/2)
Paczka dostarczona do miejsca docelowego!

Czy paczka została odebrana przez zamawiającego? (t/n): n
Paczka nie została odebrana. Czekamy na odebranie zamówienia...
Czy paczka została odebrana przez zamawiającego? (t/n): t
Paczka została odebrana. Czekamy na zapisanie książek do bazy biblioteki.
Zatwierdzanie egzemplarzy.... (1/2)
Zatwierdzanie egzemplarzy.... (2/2)
Książki są w bazie bibliotecznej!

Czy książki zostały rozłożone na półki? (t/n): n
Książki nie zostały rozłożone. Czekamy na dalsze informacje...
Czy książki zostały rozłożone na półki? (t/n): t
Książki znajdują się na odpowiednich półkach.
|
```

```
C:\Users\natal\Desktop\popr X + v
Zamówiono do biblioteki książkę Pinokio

Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać.
1. Wypożyczyć książkę czytelnikowi
2. Zwrot książki
3. Znaleźć miejsce książki w bibliotece
4. Sprawdzić brakujące egzemplarze książek
5. Dokup egzemplarze książek
6. Zamów nowe książki do biblioteki
7. Wylogować się z konta pracownika
Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: D
```

Oto jak wyglądają pliki po zamówieniu książki :

```
ksiazki.txt numer_ksiazki.txt
Plik Edytuj Wyświetl
Złodziejka książek:Markus Zusak:2005:kryminal:4:9:928
Wiedźma z Rumburaków:Jaroslav Vacek:1984:fantastyka:6:12:929
Król Artur i Rycerze Okrągłego Stołu:Sir Thomas Malory:1485:bajka:7:18:930
Dzieci z Bullerbyn:Astrid Lindgren:1947:powiesc:4:8:931
Przygoda na Wyspie Skarbów:Robert Louis Stevenson:1883:kryminal:3:6:932
Śmierć na Nilu:Agatha Christie:1937:kryminal:5:2:933
Germinal:Émile Zola:1885:powiesc:4:9:934
Podróże Guliwera:Jonathan Swift:1726:bajka:6:12:935
Złota gałąź:James George Frazer:1890:powiesc:3:7:936
Odyseja:Homery:800:p.n.e.:0:7:936
Mężczyźni, którzy nienawidzą kobiet:Stieg Larsson:2005:kryminal:4:8:938
Ciemności:Michael Ende:1979:powiesc:7:15:939
Zaginiona dziewczyna:Gillian Flynn:2012:kryminal:6:12:940
Ania z Zielonego Wzgórza:L.M. Montgomery:1908:kryminal:25:25:941
Dorotka:Zakora:2007:kryminal:15:16:942
Krzyżacy:Henryk Sienkiewicz:1900:powiesc:90:87:943
Pinokio:Dobrzyński:2003:bajka:20:15:944
```

```
ksiazki.txt numer_ksiazki.txt
Plik Edytuj Wyświetl
944
```

Krok 2 : wykonam dodatkowe zamówienie, lecz zaprzeczę. Pracuję na ostatnim pliku tekstowym.

a)

```
5. Dokup egzemplarze książek
6. Zamów nowe książki do biblioteki
7. Wylogować się z konta pracownika
Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 6

Podaj tytuł książki: aa
Podaj autora książki: ss
Podaj rok wydania książki: 2343
Wybierz rodzaj książki:
1. Bajka
2. Powieść
3. Kryminal
4. Fantastyka
5. Kulinarne
Wybierz odpowiedni numer: 4
Podaj minimalną ilość jaka musi być na stanie w bibliotece: 5
Podaj ile kupujesz egzemplarzy książki: 3
Zamówienie zostało wysłane.
```

```
C:\Users\natal\Desktop\popr: X + v
Czy zamówienie zostało opłacone? (t/n): n
Zamówienie nie zostało przyjęte. Paczka zostaje anulowana.

Process returned 0 (0x0)   execution time : 397.619 s
Press any key to continue.
```

b)

```
4. Sprawdzić brakujące egzemplarze książek
5. Dokup egzemplarze książek
6. Zamów nowe książki do biblioteki
7. Wylogować się z konta pracownika
Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 6

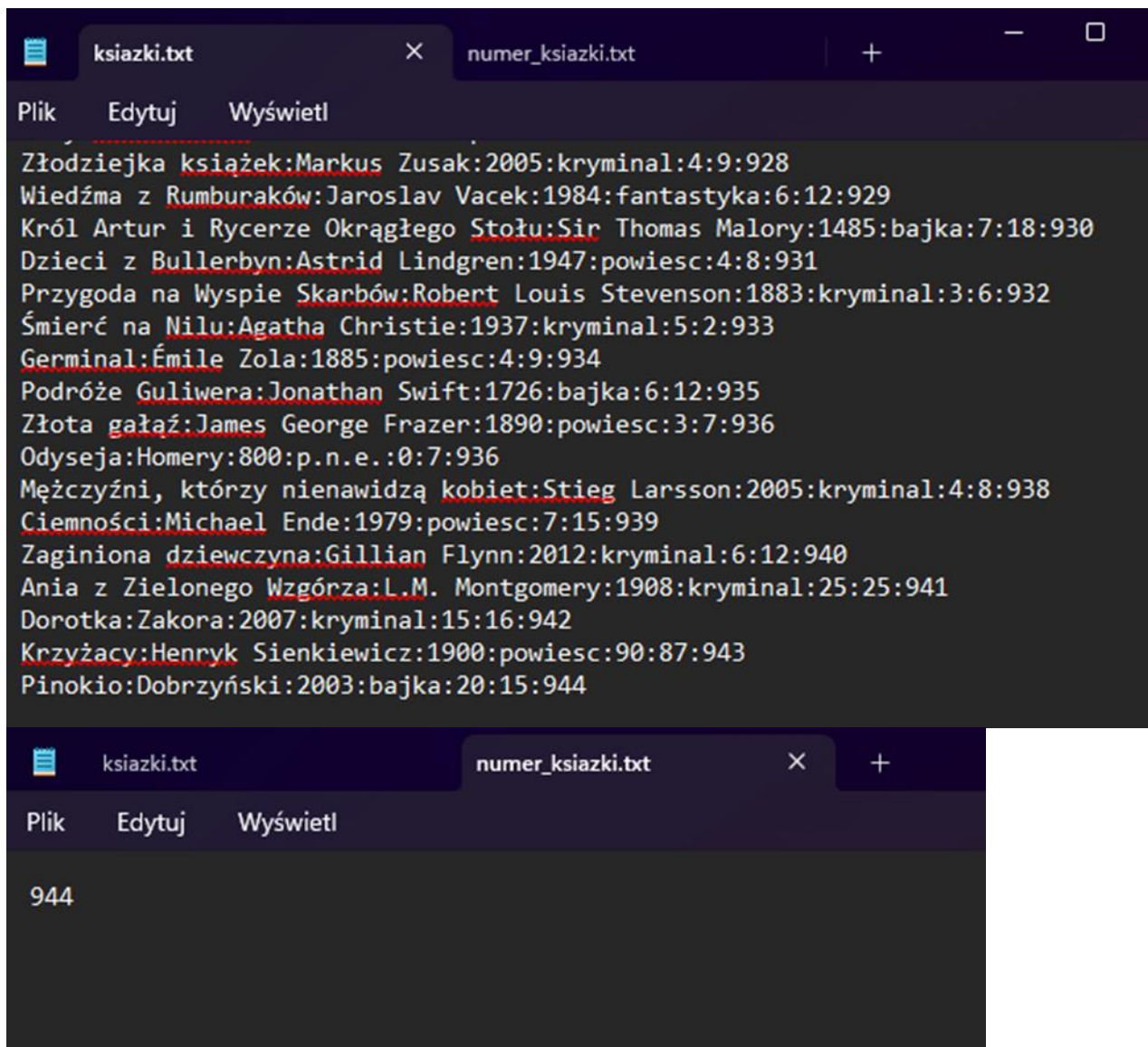
Podaj tytuł książki: asa
Podaj autora książki: ded
Podaj rok wydania książki: 1950
Wybierz rodzaj książki:
1. Bajka
2. Powieść
3. Kryminal
4. Fantastyka
5. Kulinarna
Wybierz odpowiedni numer: 5
Podaj minimalną ilość jaka musi być na stanie w bibliotece: 4
Podaj ile kupujesz egzemplarzy książki: 10
Zamówienie zostało wysłane.
```

```
C:\Users\natal\Desktop\popr: X + v
Czy zamówienie zostało opłacone? (t/n): t
Zamówienie zostało przyjęte. Rozpoczynamy realizację zamówienia...
Kompletowanie zamówienia... (1/2)
Kompletowanie zamówienia... (2/2)
Paczka jest gotowa do wysyłki!

Czy paczka została odebrana przez kuriera? (t/n): n
Paczka nie została odebrana. Czekamy na kuriera...
Czy paczka została odebrana przez kuriera? (t/n): n
Paczka nie została odebrana. Czekamy na kuriera...
Czy paczka została odebrana przez kuriera? (t/n): n
Przekroczono maksymalną liczbę prób. Paczka zostaje anulowana.

Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać.
1. Wypożyczyć książkę czytelnikowi
2. Zwrot książki
3. Znaleźć miejsce książki w bibliotece
4. Sprawdzić brakujące egzemplarze książek
5. Dokup egzemplarze książek
6. Zamów nowe książki do biblioteki
7. Wylogować się z konta pracownika
Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: |
```

Oto pliki tekstowe po tych zmianach:



## 7) WYJAŚNIENIE PLIKU ZADANIAKSIAZKACH.H

Kod ma na celu zarządzanie książkami w bibliotece, umożliwiając aktualizowanie stanu książek oraz domawianie brakujących egzemplarzy. Program wykorzystuje pliki tekstowe do przechowywania danych o książkach i obsługuje różne procesy, takie jak dodawanie książek, sprawdzanie ich stanu, oraz składanie zamówień na brakujące egzemplarze. Wykorzystuje także funkcje z pliku bazaKsiazek.h, więc muszę dołączyć plik do kod: (# include „bazaKsiazek.h”).

Wykorzystywane biblioteki:

- #include <iostream> - obsługa wejścia wyjścia
- #include <fstream> - potrzebne do pracy z plikami
- #include <string> - ułatwia pracę z tekstem
- #include <cstdlib> - służy do generowania liczb, polskich znaków,
- #include <sstream> - konwertuje różne typy danych

- `#include <chrono>` - obsługuje datę i czas
- `#include <cstdio>` - operacje wejścia wyjścia
- `#include <thread>` - praca z wieloma wątkami.

Na każdej linii z pliku odczytuje dane, używam do tego stringstream które służy do podziału linii na poszczególne pola oddzielone „:”. Odczytuje całą linię z pliku, która następnie jest przetwarzana i wykorzystuje tylko potrzebne mi dane. Przechowuje dane książek w tablicy książki i zwracam liczbę wszystkich książek.

#### Funkcja `domow_książke()`

Funkcja `domow_książke()` - umożliwia domówienie konkretnej liczby egzemplarzy do książki która już istnieje. Polega na odczycie książki z pliku „książki.txt”, podaniu przez użytkownika numeru książki którą chce dokupić oraz podaniu ilości egzemplarzy. Wszystkie te dane Użytkownik podaje w momencie kiedy program prosi o wpisanie. Następnie program sprawdza czy rzeczywiście taka książka istnieje jeśli tak. To nasz program przechodzi do funkcji która odpowiada za zamówienie książki. Jest to funkcja którą opisywałam wyżej, polegająca na opóźnieniach pracy programu. Postępuje ona zgodnie ze wcześniejszymi przeze mnie opisanymi funkcjami (zaprezentuje to poniżej). Jeśli funkcja nie odnajdzie książki o takim numerze który został wprowadzony przez użytkownika, funkcja `zloz_zamowieni()` nie zostanie uruchomiona, a wyświetli się polecenie „nie znaleziono książki o numerze...”. Ważną rzeczą jest to że jeśli chcemy uruchomić tą funkcję musimy przejść do konta pracownika zalogować się oraz wybrać opcję numer 5, która jest podpisana „Dokup egzemplarze książek”.

```

29 void domow_książke(Biblioteka& biblioteka){
30     system("chcp 65001>>null");
31
32     ifstream plik("książki.txt");
33     ofstream plikTymczasowy("książkiTym.txt");
34     string linia;
35     int nrKsiążkiPlik, iloscKsiążek;
36
37     cout << "Podaj nr książki, jaką chcesz dokupić: "; cin >> nrKsiążkiPlik;
38     cout << "Ile egzemplarzy chcesz dokupić do biblioteki: "; cin >> iloscKsiążek;
39     int znaleziono = 0;
40
41     if (!plik.is_open() || !plikTymczasowy.is_open()) {
42         cout << "Błąd przy otwieraniu plików." << endl;
43         return;
44     }
45
46     while (getline(plik, linia)) {
47         stringstream ss(linia);
48         string tytul, autor, rodzaj;
49         int rok_wydania, ilosc_min, stan, nrKsiążki;
50
51         getline(ss, tytul, ':');
52         getline(ss, autor, ':');
53         ss >> rok_wydania;
54         ss.ignore();
55         getline(ss, rodzaj, ':');
56         ss >> ilosc_min;
57         ss.ignore();
58         ss >> stan;
59         ss.ignore();
60         ss >> nrKsiążki;
61
62         if (nrKsiążki == nrKsiążkiPlik){
63             stan += iloscKsiążek;

```

```

62     if (nrKsiazki == nrKsiazkiPlik) {
63         stan += iloscKsiazek;
64         znaleziono = 1;
65     }
66
67     plikTymczasowy << tytuł << ":" << autor << ":" << rok_wydania << ":" << rodzaj << ":" << ilosc_min << ":" << stan << ":" << nrKsiazki << endl;
68 }
69
70 plik.close();
71 plikTymczasowy.close();
72
73 if (znaleziono == 1) {
74     if (remove("ksiazki.txt") != 0) {
75         cout << "Błąd przy usuwaniu pliku 'ksiazki.txt'." << endl;
76         return;
77     }
78     if (rename("ksiazkiTym.txt", "ksiazki.txt") != 0) {
79         cout << "Błąd przy zmianie nazwy pliku tymczasowego na 'ksiazki.txt'." << endl;
80         return;
81     }
82     biblioteka.zloz_zamowienie();
83     cout << "Zaktualizowano stan książki nr " << nrKsiazkiPlik << " o " << iloscKsiazek << " egzemplarzy." << endl;
84 } else {
85     remove("ksiazkiTym.txt");
86     cout << "Nie znaleziono książki o numerze " << nrKsiazkiPlik << "." << endl;
87 }
88 }

```

### Funkcja stan\_ksiazek()

Funkcja stan\_ksiazek() - odpowiedzialna jest za sprawdzenie stanu książek które są w bibliotece, by w razie braków móc domówić. System sam sprawdza wszystkie książki więc pracownik dostaje konkretną odpowiedź. Dzieje się tak, ponieważ odczytując informacje na temat stanu oraz minimalnej ilości z pliku książki.txt, program sam odejmuje te liczby otrzymując ilość brakującą. Dodatkową rzeczą jest danie pracownikowi od razu możliwości do tego by przejść do domówienia książek. Użytkownik sam decyduje czy chce od razu przejść do funkcji odpowiedzialnej za domawianie książek, czy nie.

```

118 void stan_ksiazek() {
119     ifstream plik("ksiazki.txt");
120     string linia;
121
122     while (getline(plik, linia)) {
123         Ksiazka ksiazki[MAX_KSIAZEK];
124         int liczbaKsiazek = odczytaj_ksiazki_z_pliku("ksiazki.txt", ksiazki);
125
126         string domawianie;
127         for (int i = 0; i < liczbaKsiazek; i++) {
128             int brakujace = ksiazki[i].ilosc_min - ksiazki[i].stan;
129             if (brakujace > 0) {
130                 cout << "Książka nr " << ksiazki[i].nrKsiazki << ", do poprawnej ilości w bibliotece, brakuje " << brakujace << " egzemplarzy." << endl;
131             }
132         }
133         cout << "Czy chcesz domówić książki? Wpisz tak/nie: "; cin >> domawianie;
134         plik.close();
135
136         if (domawianie == "tak") {
137             Biblioteka biblioteka;
138             domow_ksiazke(biblioteka);
139         }
140     }
141 }

```

Wynik (wykorzystam wcześniej dodaną przeze mnie książkę) :

Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać.

1. Wypożyczyć książkę czytelnikowi
2. Zwrot książki
3. Znaleźć miejsce książki w bibliotece
4. Sprawdzić brakujące egzemplarze książek
5. Dokup egzemplarze książek
6. Zamów nowe książki do biblioteki
7. Wylogować się z konta pracownika

Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 4

Książka nr 933, do poprawnej ilości w bibliotece, brakuje 3 egzemplarzy.

Książka nr 943, do poprawnej ilości w bibliotece, brakuje 3 egzemplarzy.

Książka nr 944, do poprawnej ilości w bibliotece, brakuje 5 egzemplarzy.

Czy chcesz domówić książki? Wpisz tak/nie: nie

Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać.

1. Wypożyczyć książkę czytelnikowi
2. Zwrot książki
3. Znaleźć miejsce książki w bibliotece
4. Sprawdzić brakujące egzemplarze książek
5. Dokup egzemplarze książek
6. Zamów nowe książki do biblioteki
7. Wylogować się z konta pracownika

Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: |

Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać.

1. Wypożyczyć książkę czytelnikowi
2. Zwrot książki
3. Znaleźć miejsce książki w bibliotece
4. Sprawdzić brakujące egzemplarze książek
5. Dokup egzemplarze książek
6. Zamów nowe książki do biblioteki
7. Wylogować się z konta pracownika

Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 4

Książka nr 933, do poprawnej ilości w bibliotece, brakuje 3 egzemplarzy.

Książka nr 943, do poprawnej ilości w bibliotece, brakuje 3 egzemplarzy.

Książka nr 944, do poprawnej ilości w bibliotece, brakuje 4 egzemplarzy.

Czy chcesz domówić książki? Wpisz tak/nie: tak

Podaj nr książki, jaką chcesz dokupić: 944

Ile egzemplarzy chcesz dokupić do biblioteki: 1

Zamówienie zostało wysłane.

```
C:\Users\natal\Desktop\popri x + v
Czy zamówienie zostało opłacone? (t/n): t
Zamówienie zostało przyjęte. Rozpoczynamy realizację zamówienia...
Kompletowanie zamówienia... (1/2)
Kompletowanie zamówienia... (2/2)
Paczka jest gotowa do wysyłki!

Czy paczka została odebrana przez kuriera? (t/n): n
Paczka nie została odebrana. Czekamy na kuriera...
Czy paczka została odebrana przez kuriera? (t/n): t
Paczka została odebrana przez kuriera. Kontroluj trasę paczki...
Zamówienie w drodze... (1/2)
Zamówienie w drodze... (2/2)
Paczka dostarczona do miejsca docelowego!

Czy paczka została odebrana przez zamawiającego? (t/n): t
Paczka została odebrana. Czekamy na zapisanie książek do bazy biblioteki.
Zatwierdzanie egzemplarzy.... (1/2)
Zatwierdzanie egzemplarzy.... (2/2)
Książki są w bazie bibliotecznej!

Czy książki zostały rozłożone na półki? (t/n): t
Książki znajdują się na odpowiednich półkach.
|
```

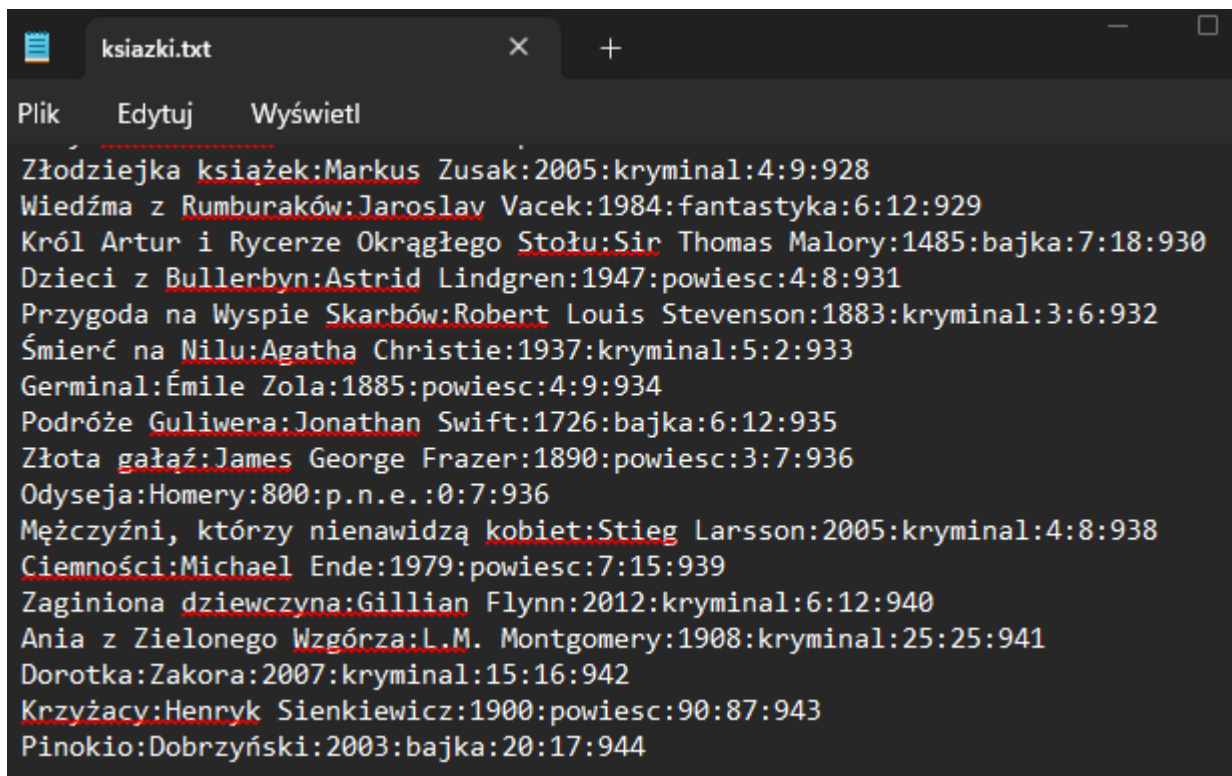
```
C:\Users\natal\Desktop\popri x + v
Zamówiono do biblioteki książkę
Zaktualizowano stan książki nr 944 o 1 egzemplarzy.

Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać.
1. Wypożyczyć książkę czytelnikowi
2. Zwrot książki
3. Znaleźć miejsce książki w bibliotece
4. Sprawdzić brakujące egzemplarze książek
5. Dokup egzemplarze książek
6. Zamów nowe książki do biblioteki
7. Wylogować się z konta pracownika
Jeśli dokonałeś wyboru, wpisz cyfrę przypisaną czynności: 4

Książka nr 933, do poprawnej ilości w bibliotece, brakuje 3 egzemplarzy.
Książka nr 943, do poprawnej ilości w bibliotece, brakuje 3 egzemplarzy.
Książka nr 944, do poprawnej ilości w bibliotece, brakuje 3 egzemplarzy.
Czy chcesz domówić książki? Wpisz tak/nie: |
```

Wynik działania w pliku książki.txt





```
Plik  Edytuj  Wyświetl
Złodziejka książek:Markus Zusak:2005:kryminal:4:9:928
Wiedźma z Rumburaków:Jaroslav Vacek:1984:fantastyka:6:12:929
Król Artur i Rycerze Okrągłego Stołu:Sir Thomas Malory:1485:bajka:7:18:930
Dzieci z Bullerbyn:Astrid Lindgren:1947:powiesc:4:8:931
Przygoda na Wyspie Skarbów:Robert Louis Stevenson:1883:kryminal:3:6:932
Śmierć na Nilu:Agatha Christie:1937:kryminal:5:2:933
Germinal:Émile Zola:1885:powiesc:4:9:934
Podróże Guliwera:Jonathan Swift:1726:bajka:6:12:935
Złota gałąź:James George Frazer:1890:powiesc:3:7:936
Odyseja:Homery:800:p.n.e.:0:7:936
Mężczyźni, którzy nienawidzą kobiet:Stieg Larsson:2005:kryminal:4:8:938
Ciemności:Michael Ende:1979:powiesc:7:15:939
Zaginiona dziewczyna:Gillian Flynn:2012:kryminal:6:12:940
Ania z Zielonego Wzgórza:L.M. Montgomery:1908:kryminal:25:25:941
Dorotka:Zakora:2007:kryminal:15:16:942
Krzyżacy:Henryk Sienkiewicz:1900:powiesc:90:87:943
Pinokio:Dobrzyński:2003:bajka:20:17:944
```

## 8) WYJAŚNIENIE PLIKU ZNAJDZKSIASZKE.H

Ten plik odpowiedzialny jest za wyszukiwanie książek w bibliotece na podstawie różnych kryteriów. Takich jak autor, rodzaj i autor. Funkcja pobiera od czytelnika bądź pracownika dane po których wybierze Użytkownik wyszukiwanie książki. Użytkownik sam wpisuje jaką opcję chce wybrać oraz sam wprowadza nazwę książki, autora, bądź rodzaju. Program wyświetla użytkownikowi w którym przedziale regałów znajduje się ten rodzaj, ponieważ biblioteka ma ustawione regały na podstawie rodzaju książek. A następnie poprzez alfabetyczne poukładanie książek wyświetla półkę na której znajduje się dana książka. Wszystko to jest podzielony na wiele funkcji. Odwołuję się także do funkcji zawartych w pliku zadaniaNaKsiazkach(), więc pamiętam o odwołaniu się do niego: #include "zadaniaNaKsiazkach.h".

Wykorzystane biblioteki:

- #include <iostream>
  - #include <string>
  - #include <algorithm>
- Funkcja porownaj\_rodzaj\_a\_titul() - wykorzystuje ją do sortowania książek początkowo po rodzaju, a następnie po tytule. Zwracam mi poprawną odpowiedź jeśli pierwszy obiekt znajduje się przed drugim

```

10
11 bool porownaj_rodzaj_a_tytul(const Ksiazka& k1, const Ksiazka& k2) {
12     if (k1.rodzaj == k2.rodzaj) {
13         return k1.tytul < k2.tytul;
14     }
15     return k1.rodzaj < k2.rodzaj;
16 }

```

- Funkcja `miejsce_ksiazki()` - wykorzystuje rodzaj i przechodząc przez warunek porównuje rodzaj do konkretnego słowa. Jeśli dane się zgadzają wypisuje w którym regale znajduje się dany rodzaj.

```

17
18 void miejsce_ksiazki(const string& rodzaj) {
19     if (rodzaj == "bajka") {
20         cout << "Ten rodzaj znajduje się na regale 1-5." << endl;
21     } else if (rodzaj == "powiesc") {
22         cout << "Ten rodzaj znajduje się na regale 6-10." << endl;
23     } else if (rodzaj == "kryminal") {
24         cout << "Ten rodzaj znajduje się na regale 11-15." << endl;
25     } else if (rodzaj == "fantastyka") {
26         cout << "Ten rodzaj znajduje się na regale 15-20." << endl;
27     } else if (rodzaj == "kulinarne") {
28         cout << "Ten rodzaj znajduje się na regale 21-25." << endl;
29     } else {
30         cout << "Nie znaleziono regału dla takiego rodzaju." << endl;
31     }
32 }
33

```

- Funkcja `oblicz_miejsce_ksiazki()` - funkcja oblicza na jakiej półce znajduje się dana książka uwzględniając jej stan w bibliotece. Na jednej półce może się znajdować 30 egzemplarzy. To właśnie tutaj odwołujemy się do wcześniej opisanej funkcji sortującej i przechodząc przez wszystkie książki obliczam miejsce na półce działaniem przedstawionym poniżej.

```

33
34 void oblicz_miejsce_ksiazki(Ksiazka ksiazki[], int liczbaKsiazek) {
35     sort(ksiazki, ksiazki + liczbaKsiazek, porownaj_rodzaj_a_tytul);
36
37     int miejsca_na_polce = 0;
38     int aktualna_polka = 1;
39
40     for (int i = 0; i < liczbaKsiazek; i++) {
41         int stan = ksiazki[i].stan;
42         while (stan > 0) {
43             int miejsca_do_zajecia = min(stan, 30 - miejsca_na_polce);
44             stan -= miejsca_do_zajecia;
45             miejsca_na_polce += miejsca_do_zajecia;
46
47             if (miejsca_na_polce == 30) {
48                 aktualna_polka++;
49                 miejsca_na_polce = 0;
50             }
51         }
52     }
53     cout << "Książka znajduje się na półce nr " << aktualna_polka << "." << endl;
54 }
55

```

- Funkcja `wyszukaj_książke_po_....()` - są tutaj 3 funkcje które różnią się jedynie rzeczą po jakiej chcemy wyszukać książkę, a jest to wcześniej przez nas wybrana opcja. Muszę wczytać książki z tablicy książki i jeśli wyszukane słowo będzie się zgadzało ze słowem wpisanym ze wcześniej pobranego pliku, wyświetli się komunikat gdzie ta książka jest w bibliotece.

```
55
56 void wyszukaj_książke_po_tytule(const string& tytul, Książka książki[], int liczbaKsiążek) {
57     sort(książki, książki + liczbaKsiążek, porównaj_rodzaj_a_tytul);
58
59     for (int i = 0; i < liczbaKsiążek; i++) {
60         if (książki[i].tytul == tytul) {
61             cout << "Znaleziono książkę: " << książki[i].tytul << " autor: " << książki[i].autor << endl;
62             miejsce_książki(książki[i].rodzaj);
63             oblicz_miejsce_książki(książki, liczbaKsiążek);
64             return;
65         }
66     }
67     cout << "Książka o tytule " << tytul << " nie została znaleziona." << endl;
68 }
69
70 void wyszukaj_książke_po_autorze(const string& autor, Książka książki[], int liczbaKsiążek) {
71     for (int i = 0; i < liczbaKsiążek; i++) {
72         if (książki[i].autor == autor) {
73             cout << "Znaleziono książkę autora: " << książki[i].autor << " tytuł: " << książki[i].tytul << endl;
74             miejsce_książki(książki[i].rodzaj);
75             oblicz_miejsce_książki(książki, liczbaKsiążek);
76             return;
77         }
78     }
79     cout << "Książki autora " << autor << " nie zostały znalezione." << endl;
80 }
81
82 void wyszukaj_książke_po_rodzaju(const string& rodzaj, Książka książki[], int liczbaKsiążek) {
83     for (int i = 0; i < liczbaKsiążek; i++) {
84         if (książki[i].rodzaj == rodzaj) {
85             cout << "Znaleziono książki w rodzaju: " << książki[i].rodzaj << endl;
86             miejsce_książki(książki[i].rodzaj);
87             return;
88         }
89     }
90     cout << "Książki o rodzaju " << rodzaj << " nie zostały znalezione." << endl;
91 }
```

### Funkcja `znajdz_książke()`

Funkcja `znajdz_książke()` - jest to nasza główna funkcje które jako jedyna z tego pliku jest wywoływana z menu\_pracownika oraz menu\_klienta. Umożliwia nam wybór, pomiędzy kryterium wyszukiwania, a następnie po odczytaniu danych przechodzi do konkretnej funkcji odpowiedzialnej za wyszukiwanie podanej nazwie.

```

92
93 void znajdz_książke() {
94     system("chcp 65001>>null");
95
96     string znajdzPo;
97     Ksiazka ksiazki[MAX_KSIAZEK];
98     int liczbaKsiazek = odczytaj_ksiazki_z_pliku("ksiazki.txt", ksiazki);
99
100     cout << "Wybierz po czym chcesz znaleźć swoją książkę w bibliotece: tytuł, rodzaj, autor" << endl;
101     cout << "Jeśli dokonałeś wybór, wpisz słowo po, którym chcesz wyszukać książkę: ";
102
103     cin.ignore();
104     getline(cin, znajdzPo);
105
106     if (znajdzPo == "tytuł") {
107         string tytuł;
108         cout << "Podaj tytuł książki: ";
109         getline(cin, tytuł);
110         wyszukaj_ksiazke_po_tytule(tytuł, ksiazki, liczbaKsiazek);
111     }
112     else if (znajdzPo == "rodzaj") {
113         string rodzaj;
114         cout << "Podaj rodzaj książki: ";
115         getline(cin, rodzaj);
116         wyszukaj_ksiazke_po_rodzaju(rodzaj, ksiazki, liczbaKsiazek);
117     }
118     else if (znajdzPo == "autor") {
119         string autor;
120         cout << "Podaj autora książki: ";
121         getline(cin, autor);
122         wyszukaj_ksiazke_po_autorze(autor, ksiazki, liczbaKsiazek);
123     }
124     else {
125         cout << "Niepoprawny wybór!" << endl;
126     }
127 }

```

Wynik:

```

Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj:
1. Znaleźć położenie książki w bibliotece
2. Sprawdzić ilość książek na swoim koncie
3. Zarejestrować się
4. Wrócić do menu głównego
Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 1

Wybierz po czym chcesz znaleźć swoją książkę w bibliotece: tytuł, rodzaj, autor
Jeśli dokonałeś wybór, wpisz słowo po, którym chcesz wyszukać książkę: tytuł
Podaj tytuł książki: Pinokio
Znaleziono książkę: Pinokio autor: Dobrzyński
Ten rodzaj znajduje się na regale 1-5.
Książka znajduje się na półce nr 20.

Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj:
1. Znaleźć położenie książki w bibliotece
2. Sprawdzić ilość książek na swoim koncie
3. Zarejestrować się
4. Wrócić do menu głównego
Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: |

```

Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj:

1. Znaleźć położenie książki w bibliotece
2. Sprawdzić ilość książek na swoim koncie
3. Zarejestrować się
4. Wrócić do menu głównego

Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 1

Wybierz po czym chcesz znaleźć swoją książkę w bibliotece: tytuł, rodzaj, autor

Jeśli dokonałeś wybór, wpisz słowo po, którym chcesz wyszukać książkę: autor

Podaj autora książki: Michael Ende

Znaleziono książkę autora: Michael Ende tytuł: Ciemności

Ten rodzaj znajduje się na regale 6-10.

Książka znajduje się na półce nr 33.

Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj:

1. Znaleźć położenie książki w bibliotece
2. Sprawdzić ilość książek na swoim koncie
3. Zarejestrować się
4. Wrócić do menu głównego

Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: |

Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj:

1. Znaleźć położenie książki w bibliotece
2. Sprawdzić ilość książek na swoim koncie
3. Zarejestrować się
4. Wrócić do menu głównego

Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 1

Wybierz po czym chcesz znaleźć swoją książkę w bibliotece: tytuł, rodzaj, autor

Jeśli dokonałeś wybór, wpisz słowo po, którym chcesz wyszukać książkę: rodzaj

Podaj rodzaj książki: bajka

Znaleziono książki w rodzaju: bajka

Ten rodzaj znajduje się na regale 1-5.

Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj:

1. Znaleźć położenie książki w bibliotece
2. Sprawdzić ilość książek na swoim koncie
3. Zarejestrować się
4. Wrócić do menu głównego

Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: |

## 9) WYJAŚNIENIE PLIKU REJESTRACJAK.H

Ten plik odpowiada tylko i wyłącznie za zarejestrowanie nowego czytelnika. Nowych klientów zapisujemy w pliku klient.txt. Program pyta użytkownika i daje mu możliwość wpisania swoich danych: imię, nazwisko oraz rok urodzenia. Do programu dodałam funkcję która ma za zadanie automatycznie sama wpisywać mi nr karty klienta by niezeszły powtórzenia, które mogły by spowodować błędy.

- isalpha: sprawdza, czy dany znak jest alfabetem. Jest zdefiniowana w pliku nagłówkowym ctype.

Wykorzystane biblioteki :

- <iostream>: Umożliwia obsługę wejścia/wyjścia

- <fstream>: Służy do operacji na plikach (odczyt i zapis danych do plików za pomocą ifstream i ofstream).
- <string>: Zawiera klasę string do pracy na znakach
- <cstdlib>: Zawiera funkcje pomocnicze, takie jak system() do obsługi polskich znaków
- <chrono>: Służy do pracy z czasem i datą, np. uzyskiwanie bieżącego roku.
- <cctype>: Zawiera funkcje do pracy z pojedynczymi znakami, np. isalpha()
- <sstream> - Umożliwia pracę ze strumieniami na łańcuchach znaków

#### Funkcja rejestracja\_klienta()

Jest to funkcja główna, która odpowiedzialna jest za proces rejestracji klienta. Zbiera dane użytkownika, sprawdza ich poprawność, a następnie zapisuje dane klienta oraz przypisuje numer karty. Użytkownik wprowadza imię, nazwisko i rok urodzenia, a program sprawdza poprawność tych danych przy użyciu funkcji sprawdz\_znaki\_alfabetyczne (dla imienia i nazwiska) oraz sprawdz\_poprawnosc\_roku (dla roku urodzenia). Zapisane jest to w pętli, która przy wpisaniu błędnych danych umożliwia poprawę tylko danego wpisu a nie całej rejestracji od początku. Uzyskujemy to dzięki podpisaniu zamiennych „0” bądź „1”, która w c++ świadczy o prawdzie bądź fałszu. Wykorzystujemy również string i int co świadczy o wykorzystywaniu równych typów danych, by uniknąć zbędnych błędów.

Numer karty jest automatycznie przypisywany na podstawie ostatniego zapisanego numeru, który przechowywany jest w pliku numer\_karty.txt i poprzedzony funkcją do tego.



```

64 void rejestracja_klienta() {
65     system("chcp 65001>>null");
66
67     string imieK, nazwiskoK;
68     int rokK;
69     int nrKarty = odczytaj_ostatni_nr_karty() + 1;
70
71     int poprawneImie = 0;
72     while (poprawneImie == 0) {
73         cout << "Podaj imie: "; cin >> imieK;
74         if (sprawdz_znaki_alfabetyczne(imieK) == 1) {
75             poprawneImie = 1;
76         } else {
77             cout << "Imie powinno zawierac tylko litery!" << endl;
78         }
79     }
80
81     int poprawneNazwisko = 0;
82     while (poprawneNazwisko == 0) {
83         cout << "Podaj nazwisko: "; cin >> nazwiskoK;
84         if (sprawdz_znaki_alfabetyczne(nazwiskoK) == 1) {
85             poprawneNazwisko = 1;
86         } else {
87             cout << "Nazwisko powinno zawierac tylko litery!" << endl;
88         }
89     }
90
91     int poprawnyRok = 0;
92     while (poprawnyRok == 0) {
93         cout << "Podaj rok urodzenia: "; cin >> rokK;
94         if (sprawdz_poprawnosc_roku(rokK) == 1) {
95             poprawnyRok = 1;
96         } else {
97             cout << "Rok urodzenia musi byc wiekszy niz 1900 i mniejszy niz biezacy rok!" << endl;
98         }
99     }
100
101     zapisz_klienta(imieK, nazwiskoK, rokK, nrKarty);
102     zapisz_numer_karty(nrKarty);
103 }

```

- Funkcja `sprawdz_znaki_alfabetyczne()` - Sprawdza, czy wszystkie znaki w ciągu znaków (dane) są literami alfabetu. Parametr zapisany jest typu `string` i przekazywany jest do funkcji przez referencje, co świadczy, że dane nie są kopiowane, a tylko się do nich odnosimy i zapewniamy dzięki `const`, że wartość dane nie będzie zmieniana. Pętla działa na każdym znaku (zmienna `c` typu `char`) w stringu `dane`. Oznacza to, że funkcja przechodzi przez każdy znak w ciągu tekstowym. Iterując po każdym znaku w `dane` i sprawdza, czy jest literą za pomocą funkcji `isalpha()`. Zwraca 1, jeśli wszystkie znaki są literami, lub 0, gdy znajdzie jakikolwiek znak, który nie jest literą alfabetu.

```

13
14 int sprawdz_znaki_alfabetyczne(const string& dane) {
15     for (char c : dane) {
16         if (!isalpha(c)) {
17             return 0;
18         }
19     }
20     return 1;
21 }
22

```

- Funkcja `sprawdz_poprawnosc_roku()` – funkcja zaciągnięta z internetu. Funkcja sprawdza poprawność roku urodzenia. Czy jest większy niż 1930 i mniejszy niż bieżący rok. Bieżący rok jest obliczany przy pomocy funkcji z biblioteki `chrono`

podobnie jak dzień wyświetlany na samym początku programu. Funkcja zwraca 1, jeśli rok jest poprawny, lub 0, gdy nie spełnia warunków.

```
23 int sprawdz_poprawnosc_roku(int rok) {
24     auto teraz = std::chrono::system_clock::now();
25     std::time_t czas = std::chrono::system_clock::to_time_t(teraz);
26     struct tm *czas_tm = std::localtime(&czas);
27     int obecny_rok = 1930 + czas_tm->tm_year;
28     return (rok > 1930 && rok <= obecny_rok);
29 }
```

- Funkcja odczytaj\_ostatni\_nr\_karty() - Polega na odczycie ostatniego numeru karty zapisanego w pliku numer\_karty.txt. Otwiera plik do odczytu, odczytuje wartość numeru karty i zamyka plik. Jeśli plik nie istnieje, przypisuje wartość 1 (początkowy numer karty).

```
31 int odczytaj_ostatni_nr_karty() {
32     ifstream plik("numer_karty.txt");
33     int ostatniNrKarty = 1;
34
35     if (plik.is_open()) {
36         plik >> ostatniNrKarty;
37         plik.close();
38     }
39     return ostatniNrKarty;
40 }
```

- Funkcja zapisz\_numer\_karty() - Zapisuje numer nowej karty do pliku numer\_karty.txt. Dzięki otwarciu pliku do zapisu, zapisuje nowy numer karty, a jeśli wystąpi błąd podczas zapisu, wyświetla komunikat o błędzie, by użytkownik wiedział, że nie został zapisany.

```
41
42 void zapisz_numer_karty(int nrKarty) {
43     ofstream plik("numer_karty.txt");
44     if (plik.is_open()) {
45         plik << nrKarty;
46         plik.close();
47     } else {
48         cout << "Błąd przy zapisywaniu numeru karty!" << endl;
49     }
50 }
51
```

- Funkcja zapisz\_klienta() - Zapisuje dane klienta (imię, nazwisko, rok urodzenia, numer karty) do pliku klient.txt. Dzięki otwarciu plik w trybie dopisywania. Zapisuje dane w formacie: imieK:nazwiskoK:rokK:nrKarty. Jeśli wystąpi błąd podczas zapisu, wyświetla odpowiedni komunikat, dzięki czemu czytelnik wie, że nie został zapisany w bazie. Dzięki const danych nie będzie można modyfikować wewnątrz funkcji, można je będzie tylko odczytywać. A użycie referencji pozwala funkcji bezpośrednio operować na oryginalnych danych przekazanych przez argumenty.

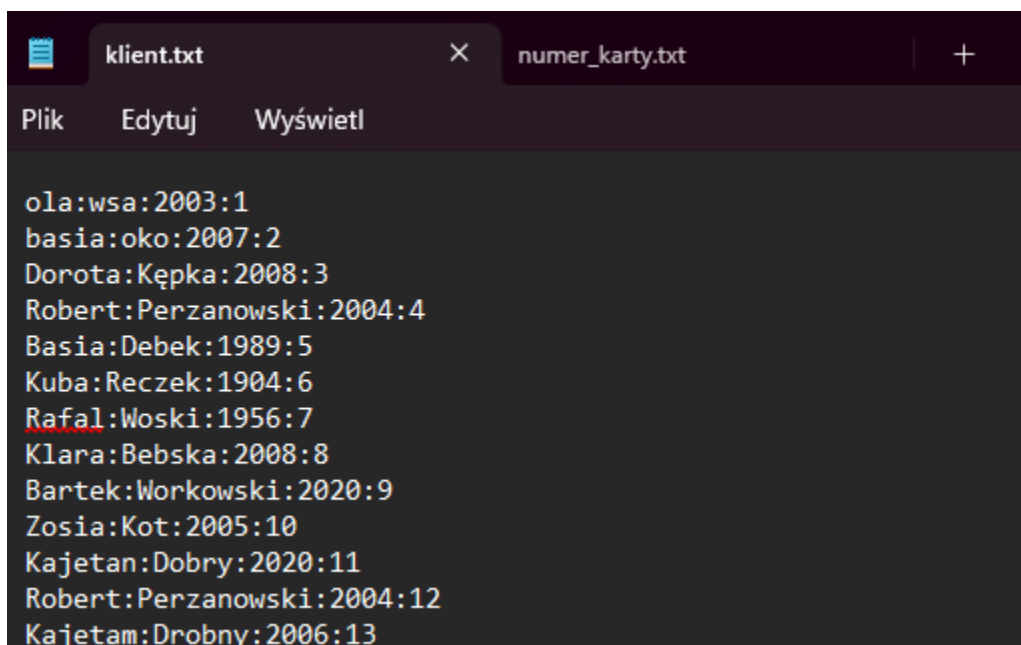
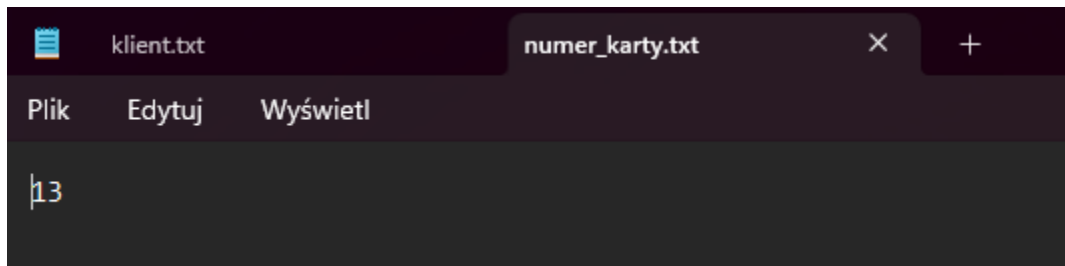


```

51
52 void zapisz_klienta(const string& imieK, const string& nazwiskoK, int rokK, int nrKarty) {
53     system("chcp 65001>>null");
54
55     ofstream plik("klient.txt", ios::app);
56     if (plik.is_open()) {
57         plik << imieK << ":" << nazwiskoK << ":" << rokK << ":" << nrKarty << endl;
58         cout << "Zostales zapisany jako czytelnik: " << imieK << " z numerem karty: " << nrKarty << endl;
59     } else {
60         cout << "Bład przy zapisywaniu danych czytelnika!" << endl;
61     }
62 }

```

Wynik pliku przed dodaniem:



Dodawanie:

Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj:

1. Znaleźć położenie książki w bibliotece
2. Sprawdzić ilość książek na swoim koncie
3. Zarejestrować się
4. Wrócić do menu głównego

Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 3

Podaj imię: 3dscr

Imie powinno zawierac tylko litery!

Podaj imię: Konrad

Podaj nazwisko: ks9/

Nazwisko powinno zawierac tylko litery!

Podaj nazwisko: Walenrod

Podaj rok urodzenia: 19038

Rok urodzenia musi byc wiekszy niz 1930 i mniejszy niz biezacy rok!

Podaj rok urodzenia: 2008

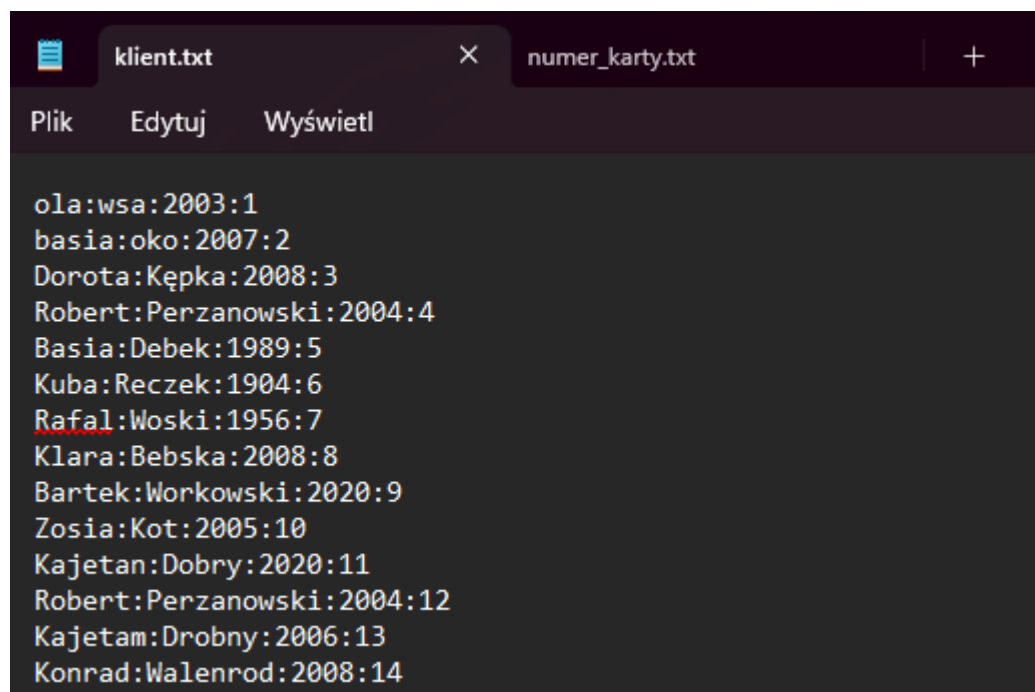
Zostales zapisany jako czytelnik: Konrad z numerem karty: 14

Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj:

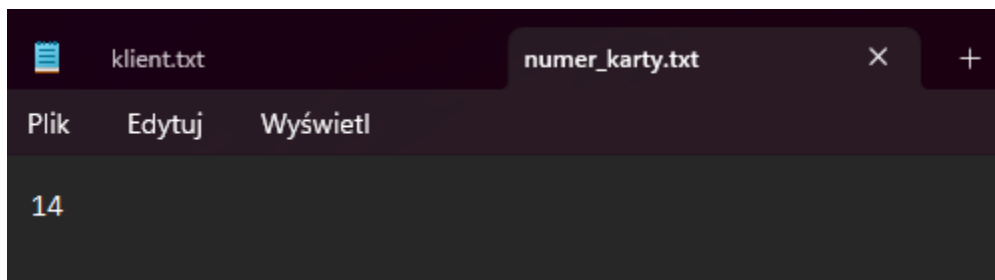
1. Znaleźć położenie książki w bibliotece
2. Sprawdzić ilość książek na swoim koncie
3. Zarejestrować się
4. Wrócić do menu głównego

Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: |

Wynik pliku po dodaniu:



```
ola:wsa:2003:1
basia:oko:2007:2
Dorota:Kępa:2008:3
Robert:Perzanowski:2004:4
Basia:Debek:1989:5
Kuba:Reczek:1904:6
Rafal:Woski:1956:7
Klara:Bebska:2008:8
Bartek:Workowski:2020:9
Zosia:Kot:2005:10
Kajetan:Dobry:2020:11
Robert:Perzanowski:2004:12
Kajetan:Drobny:2006:13
Konrad:Walenrod:2008:14
```



## 10) WYJAŚNIENIE PLIKU KLIENT.H

Zawarte w tym pliku funkcje są potrzebne do sprawnie działających dwóch funkcji, które wykorzystywane są przez pracowników oraz czytelników. Pierwszą funkcją jest `wypożycz_książke`, która ma za zadanie wypożyczenie książki danemu czytelnikowi. Dzieje się tak poprzez wprowadzenie numeru karty klienta oraz numeru książki. Każdy z tych numerów jest indywidualny i automatycznie generowany co nie doprowadzi do niepotrzebnych błędów. I jeśli mamy wypożyczenie to użytkownik także powinien mieć możliwość zwrotu. Za to działanie odpowiada kolejna funkcja `zwróć_książke()`. Która działa na tej samej zasadzie co wypożyczenie tylko zamiast pobierać egzemplarz książki ona dopisuje. Dodatkową bardzo przydatną dla czytelników funkcją jest `sprawdzenie_konta_klienta()`. Jest w stanie zweryfikować samodzielnie ile książek ma jeszcze na swoim koncie oraz dodatkowo dane wyświetlają się z datą i godziną.

Ważną rzeczą jest to, że wypożyczanie jak i zwroty książek czytelnik nie dokonać takiego zadanie, tylko pracownik biblioteki ma takie możliwości. Więc wszystkie rzeczy są wykonywane przez pracownika i na jego koncie, jako opcje nr1, 2. Jedynie czytelnik może samodzielnie sprawdzić swoje konto i ile ma na nich książek.

Plik zawiera definicję struktury klienta, która jest wykorzystywana w funkcji do przechowywania informacji o klientach. Struktura książki jest już wykorzystywana w innym pliku nagłówkowym więc jak widać, wykorzystujemy tylko odwołanie do niego dzięki `#include „bazaKsiazek.h”`. Zdefiniowana jest także stała `max_klientow`, która ustaliła maksymalną liczbę czytelników w systemie.

Wykorzystane biblioteki:

- `<iostream>`: Umożliwia obsługę wejścia/wyjścia
  - `<fstream>`: Służy do operacji na plikach
  - `<string>`: Zawiera klasę `string` do pracy na znakach
  - `<cstdlib>`: Zawiera funkcje pomocnicze, takie jak `system()` do obsługi polskich znaków
  - `<sstream>` : Umożliwia pracę ze strumieniami na łańcuchach znaków
  - `<ctime>` : Umożliwia odczytanie czasu
- Funkcja `odczytaj_klientow_z_pliku()` - Służy do odczytywania danych klientów z pliku tekstowego. Odczytuje każdy wiersz z pliku `klient.txt`, rozdziela dane klienta (imie,

nazwisko, rok urodzenia, nrKarty) i zapisuje je w tablicy struktur Klient. Zawiera dane o klientach, oraz klienci[], czyli tablica obiektów typu Klient, w której zapisane będą dane klientów są wczytane z pliku. Tworzy obiekt plik typu ifstream, który jest używany do odczytu danych z pliku. Otwiera on plik o nazwie podanej w argumencie czyli filename. Tworzy pętlę, która ma za zadanie wczytać dane z pliku do zmiennej linia. A następnie dzięki zmiennej ss, która umożliwia przetwarzanie linia jako ciąg znaków, dzieli go na poszczególne elementy oddzielone „:”. Po przetworzeniu danych klienta, obiekt klient jest zapisywany w tablicy klienci[] na pozycji wskazanej przez liczbaKlientow.

```

24 int odczytaj_klientow_z_pliku(const string& filename, Klient klienci[MAX_KLIENTOW]) {
25     ifstream plik(filename);
26     if (!plik.is_open()) {
27         cout << "Nie udało się otworzyć pliku " << filename << endl;
28         return 0;
29     }
30
31     string linia;
32     int liczbaKlientow = 0;
33     while (getline(plik, linia) && liczbaKlientow < MAX_KLIENTOW) {
34         stringstream ss(linia);
35         Klient klient;
36         getline(ss, klient.imie, ':');
37         getline(ss, klient.nazwisko, ':');
38         ss >> klient.rok_urodzenia;
39         ss.ignore();
40         ss >> klient.nrKarty;
41         klienci[liczbaKlientow++] = klient;
42     }
43
44     plik.close();
45     return liczbaKlientow;
46 }

```

- Funkcja sprawdz\_klienta() - sprawdza, czy klient o podanym numerze karty istnieje w systemie. Przeszukuje tablicę klienci[] w celu znalezienia klienta z takim numerem karty. Zwraca true, jeśli klient o danym numerze karty został znaleziony, w przeciwnym razie false.

```

48 bool sprawdz_klienta(int nrKartyKlienta, Klient klienci[MAX_KLIENTOW], int liczbaKlientow) {
49     for (int i = 0; i < liczbaKlientow; i++) {
50         if (klienci[i].nrKarty == nrKartyKlienta) {
51             return true;
52         }
53     }
54     return false;

```

- Funkcja sprawdz\_ksiazke\_dostepna() - sprawdza, czy książka o podanym numerze jest dostępna w bibliotece. Oznacza to, że książka musi istnieć oraz jej stan (liczba dostępnych egzemplarzy) musi być większy niż 0. Jeśli książka jest dostępna, jej stan jest zmniejszany o 1.

```

55
56 bool sprawdz_książkę_dostępna(int nrKsiążki, Książka książki[MAX_KSIAZEK], int liczbaKsiążek) {
57     for (int i = 0; i < liczbaKsiążek; i++) {
58         if (książki[i].nrKsiążki == nrKsiążki && książki[i].stan > 0) {
59             książki[i].stan--;
60             return true;
61         }
62     }
63     return false;
64 }

```

- Funkcja `sprawdz_wypożyczenie()` - sprawdza, czy klient o danym numerze karty już wypożyczył książkę o danym numerze. Odczytuje dane wypożyczeń z pliku `wypożyczenieK.txt`.

```

65
66 bool sprawdz_wypożyczenie(int nrKartyKlienta, int nrKsiążki) {
67     ifstream plikWypożyczenie("wypożyczenieK.txt");
68     string linia;
69     while (getline(plikWypożyczenie, linia)) {
70         stringstream ss(linia);
71         int nrKartyWyp, nrKsiążkiWyp;
72         string dataWypożyczenia;
73         ss >> nrKartyWyp;
74         ss.ignore();
75         ss >> nrKsiążkiWyp;
76         ss.ignore();
77         getline(ss, dataWypożyczenia);
78         if (nrKartyWyp == nrKartyKlienta && nrKsiążkiWyp == nrKsiążki) {
79             return true;
80         }
81     }
82     return false;
83 }
84

```

- Funkcja `zaktualizuj_książki()` - zapisuje zmiany w stanie książek (np. po wypożyczeniu książki) do pliku `książki.txt`.

```

85 void zaktualizuj_książki(Książka książki[MAX_KSIAZEK], int liczbaKsiążek) {
86     fstream plik("książki.txt", ios::in | ios::out);
87     if (plik.is_open()) {
88         plik.seekg(0, ios::beg);
89         stringstream ss;
90         for (int i = 0; i < liczbaKsiążek; i++) {
91             ss << książki[i].tytuł << ":" << książki[i].autor << ":" << książki[i].rok_wydania << ":" <<
92             << książki[i].rodzaj << ":" << książki[i].ilosc_min << ":" << książki[i].stan << ":" <<
93             << książki[i].nrKsiążki << endl;
94         }
95         plik.seekp(0, ios::beg);
96         plik << ss.str();
97         plik.close();
98     } else {
99         cout << "Błąd przy zapisywaniu książek!" << endl;
100     }
101 }

```

Funkcja `wypożycz_książkę()`

Funkcja `wypożycz_książkę()` - realizuje proces wypożyczenia książki przez klienta. Najpierw sprawdza, czy klient o podanym numerze karty istnieje, następnie sprawdza dostępność książki. Jeśli wszystko jest w porządku, książka jest wypożyczana, stan książki jest aktualizowany, a informacja o wypożyczeniu zapisywana w pliku `wypożyczenieK.txt`

```

103 void wypożycz_książke() {
104     system("chcp 65001>>null");
105
106     int nrKartyKlienta, nrKsiążki;
107     Klient klienci[MAX_KLIENTOW];
108     int liczbaKlientow = odczytaj_klientow_z_pliku("klient.txt", klienci);
109
110     Książka książki[MAX_KSIAZEK];
111     int liczbaKsiazek = odczytaj_ksiazki_z_pliku("ksiazki.txt", książki);
112
113     cout << "Podaj numer karty czytelnika: "; cin >> nrKartyKlienta;
114     if (!sprawdz_klienta(nrKartyKlienta, klienci, liczbaKlientow)) {
115         cout << "Nie znaleziono czytelnika o takim numerze karty!" << endl;
116         return;
117     }
118
119     cout << "Podaj numer książki, którą wypożyczasz: "; cin >> nrKsiążki;
120     if (!sprawdz_ksiazke_dostepna(nrKsiążki, książki, liczbaKsiazek)) {
121         cout << "Książka o numerze " << nrKsiążki << " jest niedostępna lub nie istnieje!" << endl;
122         return;
123     }
124
125     if (sprawdz_wypożyczenie(nrKartyKlienta, nrKsiążki)) {
126         cout << "Ta książka została już wypożyczona przez tego czytelnika!" << endl;
127         return;
128     }
129     zaktualizuj_ksiazki(książki, liczbaKsiazek);
130
131     ofstream plikWypożyczenieNowe("wypożyczenieK.txt", ios::app);
132     if (plikWypożyczenieNowe.is_open()) {
133         time_t now = time(0);
134         char* dt = ctime(&now);
135         plikWypożyczenieNowe << nrKartyKlienta << ":" << nrKsiążki << ":" << dt << endl;
136     } else {
137         cout << "Błąd przy zapisywaniu wypożyczenia!" << endl;
138     }
139 }

```

Funkcja sprawdz\_konto\_klienta():

Funkcja sprawdz\_konto\_klienta() - sprawdza wypożyczenia klienta na podstawie numeru karty. Odczytuje dane z pliku wypożyczenieK.txt i wyświetla informacje o książkach wypożyczonych przez klienta.

```

140
141 void sprawdz_konto_klienta() {
142     system("chcp 65001>>null");
143
144     int nrKartyKlienta;
145     cout << "Podaj numer karty: "; cin >> nrKartyKlienta;
146
147     Ksiazka ksiazki[MAX_KSIAZEK];
148     int liczbaKsiazek = odczytaj_ksiazki_z_pliku("ksiazki.txt", ksiazki);
149
150     ifstream plikWypozyczenie("wypozyczenieK.txt");
151     string linia;
152     bool znaleziono = false;
153
154     while (getline(plikWypozyczenie, linia)) {
155         stringstream ss(linia);
156         int nrKarty, nrKsiazki;
157         string dataWypozyczenia;
158
159         ss >> nrKarty;
160         ss.ignore();
161         ss >> nrKsiazki;
162         ss.ignore();
163         getline(ss, dataWypozyczenia);
164
165         if (nrKarty == nrKartyKlienta) {
166             cout << "Wypożyczono książkę nr " << nrKsiazki << " dnia: " << dataWypozyczenia << endl;
167             znaleziono = true;
168         }
169     }
170     if (!znaleziono) {
171         cout << "Ten czytelnik nie wypożyczył żadnej książki!" << endl;
172     }
173     plikWypozyczenie.close();
174 }
175

```

Funkcja zwróć\_ksiazke():

Funkcja zwroc\_ksiazke() - obsługuje zwrot książki przez klienta. Najpierw sprawdza, czy klient istnieje oraz czy książka została wypożyczona przez tego klienta. Jeśli wszystko jest w porządku, stan książki zostaje zwiększony o 1, a informacja o wypożyczeniu jest usuwana z pliku wypozyczenieK.txt. Książka jest zwrócona do systemu.



```

176 void zwroc_książke() {
177     system("chcp 65001>>null");
178
179     int nrKartyKlienta, nrKsiazki;
180     Klient klienci[MAX_KLIENTOW];
181     int liczbaKlientow = odczytaj_klientow_z_pliku("klient.txt", klienci);
182
183     Ksiazka ksiazki[MAX_KSIAZEK];
184     int liczbaKsiazek = odczytaj_ksiazki_z_pliku("ksiazki.txt", ksiazki);
185
186     cout << "Podaj numer karty czytelnika: "; cin >> nrKartyKlienta;
187     if (!sprawdz_klienta(nrKartyKlienta, klienci, liczbaKlientow)) {
188         cout << "Nie znaleziono czytelnika o takim numerze karty!" << endl;
189         return;
190     }
191
192     cout << "Podaj numer książki, którą zwracasz: "; cin >> nrKsiazki;
193     if (!sprawdz_wypozyczenie(nrKartyKlienta, nrKsiazki)) {
194         cout << "Nie znaleziono wypożyczenia tej książki dla tego klienta!" << endl;
195         return;
196     }
197
198     for (int i = 0; i < liczbaKsiazek; i++) {
199         if (ksiazki[i].nrKsiazki == nrKsiazki) {
200             ksiazki[i].stan++;
201             cout << "Książka o numerze " << nrKsiazki << " została zwrócona." << endl;
202             break;
203         }
204     }
205
206     zaktualizuj_ksiazki(ksiazki, liczbaKsiazek);
207
208     ifstream plikWypozyczenieOrig("wypozyczenieK.txt");
209     ofstream plikTemp("temp.txt");
210
211     string liniaWypozyczenia;
212     while (getline(plikWypozyczenieOrig, liniaWypozyczenia)) {
213
214         ifstream plikWypozyczenieOrig("wypozyczenieK.txt");
215         ofstream plikTemp("temp.txt");
216
217         string liniaWypozyczenia;
218         while (getline(plikWypozyczenieOrig, liniaWypozyczenia)) {
219             stringstream ss(liniaWypozyczenia);
220             int nrKartyWyp, nrKsiazkiWyp;
221             string dataWypozyczenia;
222
223             ss >> nrKartyWyp;
224             ss.ignore();
225             ss >> nrKsiazkiWyp;
226             ss.ignore();
227             getline(ss, dataWypozyczenia);
228
229             if (!(nrKartyWyp == nrKartyKlienta && nrKsiazkiWyp == nrKsiazki)) {
230                 plikTemp << liniaWypozyczenia << endl;
231             }
232         }
233         plikWypozyczenieOrig.close();
234         plikTemp.close();
235         remove("wypozyczenieK.txt");
236         rename("temp.txt", "wypozyczenieK.txt");
237
238         cout << "Wypożyczenie zostało usunięte z systemu." << endl;
239     }
240 }

```



Wynik wprowadzę zmiany na ostatnio dodanych użytkownikach oraz książkach:

```
C:\Users\natal\Desktop\poprz X + v
Witam, dziś jest: 31:01:2025
Miło cię widzieć w naszej bibliotece miejskiej w Mińsku Mazowieckim.
Wybierz cyfrę, która poprowadzi cię na odpowiednie konto:
1. Pracownik
2. Czytelnik
3. Opuść bibliotekę
Podaj wybór: 2

Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj:
1. Znaleźć położenie książki w bibliotece
2. Sprawdzić ilość książek na swoim koncie
3. Zarejestrować się
4. Wrócić do menu głównego
Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 2

Podaj numer karty: 14
Ten czytelnik nie wypożyczył żadnej książki!

Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj:
1. Znaleźć położenie książki w bibliotece
2. Sprawdzić ilość książek na swoim koncie
3. Zarejestrować się
4. Wrócić do menu głównego
Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: |
```

```
Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać.
1. Wypożyczyć książkę czytelnikowi
2. Zwrot książki
3. Znaleźć miejsce książki w bibliotece
4. Sprawdzić brakujące egzemplarze książek
5. Dokup egzemplarze książek
6. Zamów nowe książki do biblioteki
7. Wylogować się z konta pracownika
Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 1

Podaj numer karty czytelnika: 14
Podaj numer książki, którą wypożyczasz: 944

Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać.
```

Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj:

1. Znaleźć położenie książki w bibliotece
2. Sprawdzić ilość książek na swoim koncie
3. Zarejestrować się
4. Wrócić do menu głównego

Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 2

Podaj numer karty: 14

Wypożyczono książkę nr 944 dnia: Fri Jan 31 00:04:36 2025

Wypożyczono książkę nr 944 dnia:

Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj:

1. Znaleźć położenie książki w bibliotece
2. Sprawdzić ilość książek na swoim koncie
3. Zarejestrować się
4. Wrócić do menu głównego

Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: |

Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać.

1. Wypożyczyć książkę czytelnikowi
2. Zwrot książki
3. Znaleźć miejsce książki w bibliotece
4. Sprawdzić brakujące egzemplarze książek
5. Dokup egzemplarze książek
6. Zamów nowe książki do biblioteki
7. Wylogować się z konta pracownika

Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 4

Książka nr 933, do poprawnej ilości w bibliotece, brakuje 3 egzemplarzy.

Książka nr 943, do poprawnej ilości w bibliotece, brakuje 3 egzemplarzy.

Książka nr 944, do poprawnej ilości w bibliotece, brakuje 4 egzemplarzy.

Czy chcesz domówić książki? Wpisz tak/nie: nie

Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać.

1. Wypożyczyć książkę czytelnikowi

6. Zamów nowe książki do biblioteki

7. Wylogować się z konta pracownika

Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 2

Podaj numer karty czytelnika: 14

Podaj numer książki, którą zwracasz: 944

Książka o numerze 944 została zwrócona.

Wypożyczenie zostało usunięte z systemu.

Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać.

1. Wypożyczyć książkę czytelnikowi
2. Zwrot książki
3. Znaleźć miejsce książki w bibliotece
4. Sprawdzić brakujące egzemplarze książek
5. Dokup egzemplarze książek
6. Zamów nowe książki do biblioteki
7. Wylogować się z konta pracownika

Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 4

Książka nr 933, do poprawnej ilości w bibliotece, brakuje 3 egzemplarzy.

Książka nr 943, do poprawnej ilości w bibliotece, brakuje 3 egzemplarzy.

Książka nr 944, do poprawnej ilości w bibliotece, brakuje 3 egzemplarzy.

Czy chcesz domówić książki? Wpisz tak/nie: nie

Oto konto pracownika. Wybierz cyfrę, która przekieruje cię do bazy z odpowiednim zadaniem, które chcesz wykonać.

1. Wypożyczyć książkę czytelnikowi

Wybierz cyfrę, która poprowadzi cię na odpowiednie konto:

1. Pracownik
2. Czytelnik
3. Opuść bibliotekę

Podaj wybór: 2

Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj:

1. Znaleźć położenie książki w bibliotece
2. Sprawdzić ilość książek na swoim koncie
3. Zarejestrować się
4. Wrócić do menu głównego

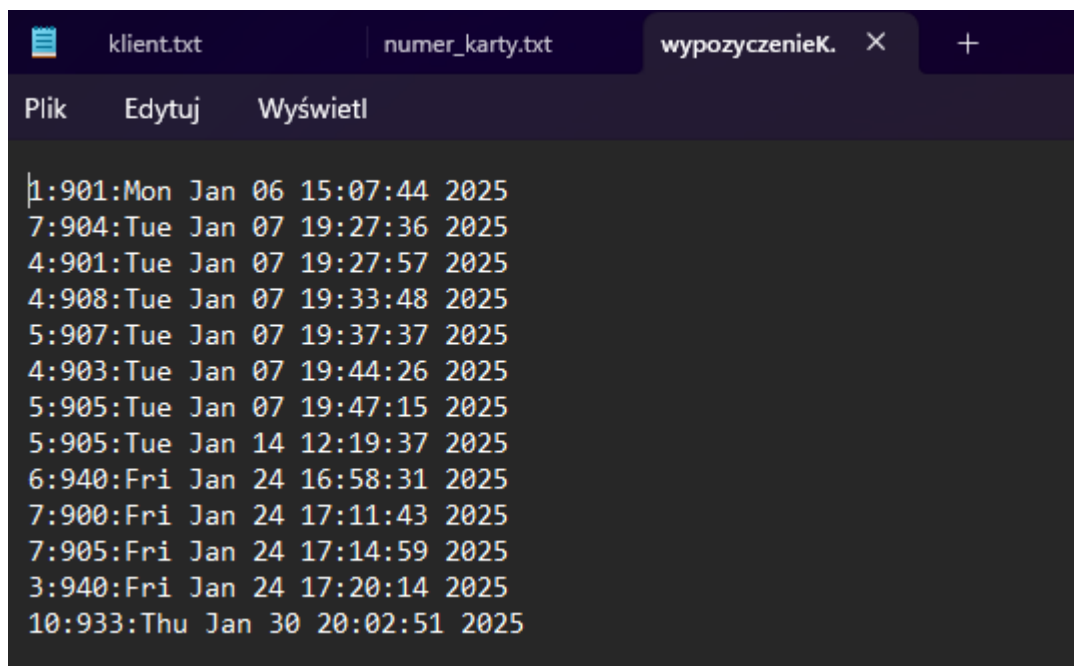
Jeśli dokonałeś wyboru, wpisz cyferkę przypisaną czynności: 2

Podaj numer karty: 14

Ten czytelnik nie wypożyczył żadnej książki!

Witaj czytelniku. Wybierz cyfrę, która odpowiada twojemu zadaniu na dzisiaj:

1. Znaleźć położenie książki w bibliotece
2. Sprawdzić ilość książek na swoim koncie



The screenshot shows a terminal window with three tabs: 'klient.txt', 'numer\_karty.txt', and 'wypozyczenieK.'. The 'wypozyczenieK.' tab is active. Below the tabs is a menu bar with 'Plik', 'Edytuj', and 'Wyświetl'. The main area of the terminal displays a list of transactions, each on a new line. Each line contains a card number, a day of the week, a month, a day, a time, and a year, separated by colons and spaces. The transactions are as follows:

Card Number	Day	Month	Day	Time	Year
1:901	Mon	Jan	06	15:07:44	2025
7:904	Tue	Jan	07	19:27:36	2025
4:901	Tue	Jan	07	19:27:57	2025
4:908	Tue	Jan	07	19:33:48	2025
5:907	Tue	Jan	07	19:37:37	2025
4:903	Tue	Jan	07	19:44:26	2025
5:905	Tue	Jan	07	19:47:15	2025
5:905	Tue	Jan	14	12:19:37	2025
6:940	Fri	Jan	24	16:58:31	2025
7:900	Fri	Jan	24	17:11:43	2025
7:905	Fri	Jan	24	17:14:59	2025
3:940	Fri	Jan	24	17:20:14	2025
10:933	Thu	Jan	30	20:02:51	2025