**Uzupełnij klasę Pociąg:**

* Konstruktor powinien inicjalizować obiekt wartościami domyślnymi np. 0 przy parametrach których w początkowej części programu nie będziemy używać np. accesWIFI=0
* Zastanów się nad użytymi typami np. std::string get\_accesWIFI może być typem „bool” i być wartością false jeżeli pociąg nie posiada wifi albo true jeżeli posiada. I tak z resztą metod… (po co nam maxSpeed pociągu do planera podróży???)
* Dodaj zmienną typu int zawierającą dni, w których pociąg kursuje
* Dodaj komentarze do klasy ułatwiające identyfikację metod i zmiennych współpracownikom

**Uzupełnij klasę Route:**

* Popraw składnię klasy np. przed słowem public powinna być klamra otwierająca klasę, poszukaj też innych błędów składniowych
* Dodaj do klasy dziedziczenie metod klas Train i Station (aktualnie używasz ich w klasie bez dziedziczenia)
* Konstruktor klasy powinien tylko i wyłącznie inicjalizować przekazanymi zmiennymi (cały kod z odczytywaniem pliku jest niepotrzebny, tym zajmują się funkcje, konstruktor tylko inicjalizuje)
* Zastanów się nad parametrami jakie powinna zawierać klasa Route, np. w kontenerze std::vector<int> można przechowywać godziny pryjazdu i odjazdu, zauważ że liczba stacji danego pociągu może być różna. Wektorze stringów nazwy stacji na których pociąg się zatrzymuje, itp.
* Zapoznaj się z plikiem ciapongi.txt i stacje.txt, połączenie ich dwóch pozwala stworzyć całą klasę Route, przykład: patrzysz na linijkę NAME w pliku ciapongi.txt więc tworzysz w klasie zmienną która dziedziczy z klasy pociąg nazwę i zostanie zainicjalizowana przez konstruktor przekazanym stringiem. Przykład2: patrzysz na linijkę STATION to dzidziczysz z klay Station jej numer i i wrzucasz do wektora, przykład 3:patrzysz na linijkę IN widzisz że godzina jest 23 a żadna z klas nie ma takiego parametru to tworzysz nowy wektor intów do którego będziesz wrzucać godziny przyjazdów, ale pamiętaj że to nie konstruktor ma czytać z pliku gdzie jest godzina przyjazdu tylko będzie mu przekazany wektor zawierający już te informacje który stworzy jakaś inna funkcja zajmująca się odczytem z pliku! (analogicznie wektory zawierające minutę przyjazdu, godzinę odjazdu, minutę odjazdu)
* Dodaj komentarze do klasy ułatwiające identyfikację metod i zmiennych współpracownikom

**Napisz funkcję, która odczyta dni kursowania z pliku ciapongi.txt:**

* Do funkcji przekazujemy nazwę pociągu(string), ma go wyszukać w pliku i zwrócić wartość w postaci liczby (np.1010110)(sprawdź jaki typ w klasie Train)
* Pamiętaj, że każda odczytana linijka z pliku jest typu string i trzeba ją przekonwertować na odpowiedni typ.
* Zwróć uwagę na błędy, jakie mogą się zdarzyć podczas wywołania funkcji i zapobiegnij im
* Funkcja deklarację powinna mieć w pliku hpp a definicję w ciapong.cpp
* Opatrz funkcję stosownymi komentarzami ułatwiającymi identyfikację współpracownikom

**Napisz funkcję, która odczyta klasę pociągu(tlk/ic/eip/eic) z pliku ciapongi.txt:**

* Do funkcji przekazujemy nazwę pociągu(string), ma go wyszukać w pliku i zwrócić wartość w postaci cyfry odpowiadającej klasie (patrz opis ciapongi.txt)(a typ zwracany w klasie train)
* Pamiętaj, że każda odczytana linijka z pliku jest typu string i trzeba ją przekonwertować na odpowiedni typ.
* Zwróć uwagę na błędy, jakie mogą się zdarzyć podczas wywołania funkcji i zapobiegnij im
* Funkcja deklarację powinna mieć w pliku hpp a definicję w ciapong.cpp
* Opatrz funkcję stosownymi komentarzami ułatwiającymi identyfikację współpracownikom

**Napisz funkcję, która znajdzie wskazaną trasę:**

* Do funkcji przekazujemy dwa obiekty typu Station(jedna stacja początkowa, z której ktoś chce jechać druga końcowa)
* Funkcja wykorzystuje funkcję find\_station żeby znaleźć numer odpowiadający danej stacji
* Następnie w pliku ciapongi.txt przeszukuje pociągi i znajduje trasy pociągów zawierające daną relację (bez patrzenia na dni kursowania, klasę czy godziny przyjazdu, sprawdzeniem tego zajmie się inna funkcja, ta ma tylko wyszukiwać, jaki pociąg kursuje na danej trasie i zwracać daną trasę)
* Funkcja zwraca obiekt typu Route( musi zawierać nazwę pociągu, numer wszystkich stacji, godziny odjazdu przyjazdu)
* Pamiętaj, że każda odczytana linijka z pliku jest typu string i trzeba ją przekonwertować na odpowiedni typ.
* Zwróć uwagę na błędy, jakie mogą się zdarzyć podczas wywołania funkcji i zapobiegnij im
* Funkcja deklarację powinna mieć w pliku hpp a definicję w ciapong.cpp
* Opatrz funkcję stosownymi komentarzami ułatwiającymi identyfikację współpracownikom

**Napisz funkcję która przefiltruje zapytanie(this is hard):**

* Funkcja otrzymuje dwa obiekty typu Station(relacja- skąd,dokąd),dzień w którym pasażer chce jechać w formie “0100000”-(wtorek), godzinę i minutę odjazdu(int), klasę pociągu(int)
* Przy pomocy “funkcji znajdującej trasy,funkcji odczytującej klasę pociągu, funkcji odczytującej dni kursowania” odfiltrować te które pasują do zapytania
* Zwraca odpowiedni obiekt
* Zwróć uwagę na błędy, jakie mogą się zdarzyć podczas wywołania funkcji i zapobiegnij im
* Funkcja deklarację powinna mieć w pliku hpp a definicję w ciapong.cpp
* Opatrz funkcję stosownymi komentarzami ułatwiającymi identyfikację współpracownikom

**Interfejs użytkownika:**

* w pliku main.cpp napisz interfejs użytkownika pobierający relację(dwa obiekty typu station, godzinę odjazdu,minutę odjazdu, klasę pociągu
* zwróć uwagę na WSZYSTKIE możliwe błędy popełnione przez użytkownika podczas wprowadzania danych
* zrób to w taki sposób aby współgrało z powyższymi funkcjami
* Opatrz wszystko stosownymi komentarzami ułatwiającymi identyfikację współpracownikom

pliki znajdują się w ścieżce: ...\Program ciapongi\cmake-build-debug

**PLIK CIAPONG.TXT-opis**

**NAME**<-znacznik nazwy pociągu

**Janusz**<-nazwa pociągu

**CLASS**<-znacznik klasy pociągu

**2**<-numer klasy pociągu 1 - TLK, 2 - IC, 3 - EIC, 4 - EIP

**DAY**<-znacznik dnia w którym pociąg kursuje

**1111111**<- cyfry po kolei oznaczające dzień tygodnia, 1 - kursuje, 0 - nie, przykład: 1111100 oznacza że kursuje od pn do pt, ale w weekendy nie

**STATION**<-znacznik nowej stacji

**2**<-numer stacji (odpowiada stacji z pliku stacje.txt)

**IN**<-znacznik przyjazdu na stację

**23**<-godzina przyjazdu na stację

**56**<-minuta przyjazdu na stację

**OUT**<-znacznik odjazdu ze stacji

**11**<-godzina odjazdu ze stacji

**12**<-minuty odjazdu ze stacji

**STATION**<-kolejna stacja na trasie pociągu i tak dalej

**3**

**IN**

**23**

**12**

**OUT**

**12**

**12**

**END\_STATION**<-znacznik końca trasy i jednocześnie końca jednego rekordu w bazie

**NAME**<-znacznik kolejnego pociągu(analogicznie)

**Janusz\_expres**

**CLASS**

**1**

**DAY**

**1010101**

**STATION**

**5**

**IN**

**12**

**12**

**OUT**

**12**

**12**

**STATION**

**2**

**IN**

**21**

**55**

**OUT**

**11**

**43**

**END\_STATION**

**PLIK STACJE.TXT-opis**

**1**<-znacznik numeru stacji w bazie „Kraków Główny”

**Krakow\_Glowny**<-nazwa stacji

**2**<-znacznik kolejnej stacji…

**Lublin**

**3**

**Tarnow**

**4**

**Warszawa\_Centralna**

**5**

**Wachock**