Wymagania minimalne na zaliczenie z przedmiotu Języki Obiektowe na ocenę 3.

UWAGA: Podstawą zaliczenia nie jest pokaz jakiegoś działającego programu lecz obrona weryfikująca 100% - owe autorstwo i wiedzę na temat struktury i tworzenia programu oraz systematyczne prezentowanie stanu projektu na kolejnych zajęciach, zatem obowiązkowa obecność na zajęciach z pokazem dotychczasowych postępów. Prace „przyniesione" jedynie na ostatnie zajęcia nie będą sprawdzane.

Podczas „obrony" projektu będą zadawane pytania weryfikujące autorstwo i wiedzę.

Dzieła zawierające nieautorski wkład intelektualny zainicjują postępowanie dyscyplinarne.

Można wykorzystywać notatki, kody stworzone na zajęciach laboratoryjnych z całkowitym zrozumieniem ich treści.

Stworzyć aplikację napisaną **w całości** w języku C++, posiadającą następujące cechy i funkcjonalności:

1. Aplikacja może być konsolowa lub okienkowa
2. **Minimum dwie** klasy bezpośrednio związane z gromadzeniem i przetwarzaniem danych (nie wizualizacją itp.)
3. Klasa opisująca podstawowy obiekt musi posiadać **minimum 6 składowych pól danych** (pól rekordu – cech obiektu), z których co najmniej jedna musi być numeryczna i co najmniej jedna tekstowa. Przykładami klas ogólnych mogą być: osoba, pracownik, student, towar, komputer, mebel, pojazd, samochód, samolot, gra itp.
4. Powyższa (powyższe, gdy jest ich więcej) klasa umożliwiają dostęp do danych jedynie poprzez metody (setery, getery, …), a nie przez interfejs użytkownika.
5. Kolejna klasa zawiera strukturę(y) danych (kontener) gromadzącą obiekty zdefiniowane przez klasy powyższe oraz metody do zarządzania tym kontenerem danych. Przykładami takich klas mogą być: kadraPracownicza, grupaStudencka, sklepRowerowy, autosalon, przewoźnik, itp.
6. Tworzenie nowego, czystego pliku (z potwierdzeniem).
7. Otwieranie/czytanie z pliku wszystkich danych do pamięci (kontenera danych).
8. Zapis do pliku wszystkich danych z pamięci (z kontenera danych).
9. Przeglądanie danych o 1 w przód i w tył.
10. Aktualizacja pojedynczego – aktualnie przeglądanego – elementu po potwierdzeniu przez użytkownika.
11. Usuwanie pojedynczego – aktualnie oglądanego – elementu po potwierdzeniu przez użytkownika.
12. Dopisywanie pojedynczego elementu na koniec kontenera danych w pamięci.
13. Przy dopisywaniu i aktualizacji obowiązuje podstawowa walidacja (zabezpieczenie przed np. cenami<=0 lub zbyt dużymi cenami, wiekiem pracownika <18 lub zbyt wysokim wiekiem itp.).

14. Wyszukiwanie z możliwością przeglądania wyników o 1 w przód i 1 w tył. **Wyszukiwanie minimum według 2 niezależnych kryteriów** skierowanych do **konkretnych** pól:
    a. Tekstowego

UWAGA: jeśli szukamy konkretnego **nazwiska** np. Seweryn, to program nie szuka imienia Seweryn.

b. numerycznego z **przedziałem szukanych wartości**
UWAGA: jeśli szukamy konkretnie **długości** od 100 do 200 to program nie może sprawdzać również szerokości, wysokości czy masy itp.

15. Sortowanie rosnąco i malejąco według 2 niezależnych kryteriów:
    a. Wybranego tekstowego
    b. Wybranego numerycznego
16. Wyjście z programu z potwierdzeniem.
17. Do prezentacji i obrony plik z danymi musi zawierać minimum 12 sensownie przygotowanych danych, niektóre mogą być podobne by lepiej zaprezentować wyszukiwanie.
18. Przygotować **krótki** dokument opisujący tematykę i możliwości projektu. Strona tytułowa pliku zawiera: tytuł, imię i nazwisko autora, grupę studencką, datę ukończenia projektu.

Obowiązkowa obecność na zajęciach z pokazem dotychczasowych postępów. Prace „przyniesione" jedynie na ostatnie zajęcia nie będą sprawdzane.

Podczas „obrony" projektu będą zadawane pytania weryfikujące autorstwo i wiedzę.

Dodatkowy wkład projektowy, własna inwencja lub zastosowanie wybranego/wybranych rozszerzeń opisanych poniżej lub własnych pozwoli uzyskać wyższą ocenę.

Przykładowy wybór dodatków (lepszy projekt lub dodatkowe funkcjonalności) pozwalających uzyskać wyższą ocenę:

1. Dołożenie klasy dziedziczącej/rozszerzającej (lub więcej takich klas) względem klasy podstawowej. Można zatem utworzyć klasę bardziej ogólną, która przyda się w przyszłości oraz dziedziczącą od niej (lub parę dziedziczących), która precyzuje cechy przyszłych obiektów. Przykłady: od klasy ***osoba*** zawierającej ogólne cechy (np. imię, nazwisko, PESEL…) mogłyby dziedziczyć klasy ***student, pracownik, pacjent, klient*** (każda z dziedziczących klas ma inne cechy szczególne) mogą być: pracownik lub student itp., od klasy towar mogłyby dziedziczyć klasy ***auto, gra, komputer, mebel*** (tu znów każda z dziedziczących klas ma inne cechy szczególne).
2. Praca z wieloma plikami (wybór, archiwizacja, zapisz jako, zapisywanie wyników wyszukiwania…)
3. Zaznaczanie / wyłączanie zaznaczenia elementów do usunięcia, archiwizacji...
4. Usuwanie grupowe elementów wcześniej zaznaczonych lub na podstawie określonych kryteriów (może być usuwanie elementów wcześniej wyszukanych)
5. Tworzenie kosza z obsługą: przegląd danych o 1 w przód i w tył, przywracanie oglądanego lub grupy np. zaznaczonych, trwałe usunięcie oglądanego, opróżnienie kosza.
6. Zaawansowane kryteria wyszukiwawcze (z operatorami logicznymi, dokładne / przybliżone…).
7. Wyszukaj i zamień – automatyczna modyfikacja dla spełnionego kryterium dla wszystkich danych lub po potwierdzeniu dla wybranych.
2. Zastosowanie / wyłączanie filtrów (niezależnie od wyszukiwania).

3. Zaawansowana walidacja np. sprawdzanie poprawności dat, „martwe klawisze” nieliterowe podczas wprowadzania imion itp., inne zabezpieczenia.
4. Użytkownicy z różnymi uprawnieniami…
5. Wydruki.
6. Eksport pliku do innych formatów.
7. „Baza” z powiązanymi relacyjnie tabelami
8. Hasło dostępu do operacji modyfikujących (dla pkt. 10, 11, 12 lub inne pokrewne) zapisywane w pliku tekstowym w postaci niejawnej.

Oczywiście własne pomysły na wzbogacenie projektu są jak najbardziej dopuszczalne.