



Zadanie 1

- Napisz program, do którego, jako argument możemy przekazać imię i wypisze
 - Przekaż imię jako parametr do metody `Console.WriteLine`



Zadanie 2

- Napisz program, do którego możemy wprowadzić długości trzech boków trójkąta
 - Użyj metody `int.Parse`
- Sprawdź, czy mogą one stworzyć trójkąt.



Zadanie 3

- Napisz program, który sprawdza czy dana liczba jest liczbą pierwszą
 - Wydziel nową metodę, która zwraca wartość bool
 - Argumentem wejściowym jest rozpatrywana liczba



Zadanie 4

- Napisz program, który znajduje najmniejszą liczbę Fibonacciego większą od:
 - 1000
 - parametru z args
 - Wypisz na końcu wszystkie kolejne liczby Fibonacciego (1,1,2,3,5,8,13,21,34,55)



Zadanie 5

- Sprawdź ile dni minęło od dnia Twoich urodzin do teraz.
 - Wynik wypisz do konsoli
 - Policz czas również w godzinach.



Zadanie 6

- Napisz program, który na podstawie następującego tekstu „Lorem ipsum dolor sit amet, consectetur adipiscing elit.”:
 - Zlicza liczbę wyrazów (Split)
 - Wypisuje fragment tekstu do przecinka (bez przecinka) (IndexOf, Substring)
 - wypisuje cały tekst za przecinkiem (bez przecinka)
 - Wypisuje tylko trzeci wyraz (Split)
 - Łączy w jeden string co drugi wyraz (Split)
 - Zlicza wystąpienie litery ‘e’ (foreach or Split, or IndexOf and Substring)



Zadanie 7

- Porównaj prędkość działania:
 - Konkatenacji stringów
 - StringBuilder'a



Zadanie 8

- Napisz klasę Robot z następującymi trzema zmiennymi instancji:
 - Name(string), Age(ushort) i IsOn(bool), nadaj im Gettery i Settery.
 - Pozwól użytkownikowi na przypisywanie wartości Name oraz Age w konstruktorze, gdy tworzony jest obiekt Robot. Zapewnij również istnienie konstruktora domyślnego
 - właściwość IsOn jest inicjowana domyślnie z wartością true
 - Dodaj nowa metode SayHi() która, jeśli właściwość IsOn jest true wypisuje na konsolę "Say Hi {Name}"



Zadanie 9

- Napisz klasę RainFall zawierającą jednowymiarową tablicę z 12 elementami reprezentującymi miesięczne pomiary opadów deszczu. Uwzględnij w niej następujące cechy:
 - Klasa powinna udostępniać użytkownikom metodę GetMonthlyRainFall na dostęp do elementów tablicy poprzez indeks typu int, ale miesiąc pierwszy powinien być dostępny przez indeks 1
 - Dołącz właściwość o nazwie Average, która policzy średni roczny opad miesięczny
 - Napisz metodę AddRainFall, która dodaje opad do danego miesiąca
 - Napisz metodę ImportRainFall która przyjmuje jako parametr typ RainFall, która dodaje do obecnej instancji wszystkie 12 opadów z przekazanej zmiennej



Zadanie 10

- Napisz cztery klasy o nazwach Employee, Secretary, Director oraz Programmer.
 - klasa Employee jest klasą bazową zawierającą właściwość Id, Name oraz metodę CalculateSalary, która wypisuje „Wypłacam dla [nazwaOsoby]”
 - klasy pochodne nadpisują CalculateSalary() dopisując nową linię z wysokością pensji „Kwota: [kwota]”
 - ponadto niech Director zawiera metodę GetBonus() (Niech wypisze na konsolę „Wypłacam bonus.”)
 - Napisz metodę, która tworzy po 2 obiekty Secretary oraz Programmer i 1 obiekt Director. Zapisz je w dowolnej kolekcji
 - Napisz metodę, która jako parametr przyjmuje tę kolekcję i wykonuje dla wszystkich obiektów metodę CalculateSalary(), a dla Director również GetBonus();



Zadanie 11

- Przesłoń metodę ToString() tak, aby wszystkie klasy dziedziczące po Employee wypisywały treść:
 - “My name is {Name}, Id: {Id}”



Zadanie 12

- Przeciąż metodę Equals() tak, aby:
 - zwracała true, jeżeli porównywane obiekty typu Employee mają takie samo Id
 - W innych przypadkach powinna zwracać false
 - Pamiętaj o przeciążeniu GetHashCode()!



Zadanie 13

- Napisz klasę pozwalającą na wczytanie konfiguracji z pliku konfiguracyjnego aplikacji:
 - Elementy konfiguracji aplikacji to:
 - poziom logowania (enum – Debug, Warning, Error)
 - język aplikacji (CultureInfo)
 - Klasa posiada właściwości z getterami i setterami
 - Właściwości są inicjalizowane podczas tworzenia instancji klasy (w konstruktorze)



Zadanie 14

- Dla Klasy przykładowej FamilyCar napisz nowy typ wyjątku CapacityExceededException:
 - Do wyjątku przekaz informacje o maksymalnej pojemności, pojemności aktualnej i wartości, którą próbowano przypisać
 - Utwórz nowy obiekt typu FamilyCar i wywołaj metodę LoadTrunk, z parametrem który powinien wyzwoić nowy wyjątek
 - Przechwyc ten wyjątek w metodzie, w której inicjalizujemy obiekt
 - Wypisz informację z wyjątku na konsolę



Zadanie 15

- **Napisz metodę rozszerzającą dla:**
 - Klasy string pozwalającej liczyć wyrazy w tekście
 - Klasy string zliczającą wystąpienie danej litery w tekście
 - Klasy FamilyCar, która będzie wykonywała metodę LoadTrunk dla każdego elementu z listy przekazanej jako parametr. Metoda zwraca true jeśli, wszystkie elementy udało się spakować



Zadanie 16a

- Wykorzystaj klasę Users, do następujących operacji wykorzystujących LINQ (pobierz kolekcję za pomocą `CreateCollection.GetUsers()`):
 - Sprawdź czy kolekcja posiada jakikolwiek element (`Any`)
 - Sprawdź czy kolekcja posiada jakiegokolwiek aktywnego użytkownika (`Any`)
 - Policz wszystkie nie nullowe elementy (`Where`, `Count`)
 - Wypisz wszystkich użytkowników, których imię zaczyna się na literę "M" (`Where`)
 - wyciągnij 5 pierwszych użytkowników posortowanych po imieniu (`Take`, `OrderBy`)
 - wypisz drugą piątkę (`Take`, `Skip`)
 - Wypisz wszystkie imiona bez powtórzeń (`Select + Distinct`, `DistinctBy`)
 - Wypisz wszystkie imiona, które się powtarzają (`GroupBy`)
 - Wypisz wszystkich użytkowników, których się powtarzają (`GroupBy`, `SelectMany`)



Zadanie 16b

- Wykorzystaj klasę Users, do następujących operacji wykorzystujących LINQ (pobierz kolekcję za pomocą `CreateCollection.GetUsers()`):
 - Wypisz tylko obiekty typu SuperUser (OfType)
 - Wypisz wszystkich użytkowników, którzy są aktywnymi administratorami
 - Pobierz tylko jednego użytkownika, który jest administratorem (FirstOrDefault).
 - Utwórz obiekt Dictionary<string, int> gdzie kluczem jest imię użytkownika, a wartością liczba użytkowników o takim samym imieniu (GroupBy, ToDictionary)