

po-2021-22L-wyklad-kmolenda

Uczestnicy

Kompetencje

Oceny

Zaliczenie przedmiotu

Podręczniki, kurzy, materiały

Repetytorium

Konwencje kodowania w C# i dokumentowanie kodu

System typów języka C# (platformy .Net)

Modelowanie obiektowe w C#

Well formed types

Kokpit

Strona główna

Kalendarz

Prywatne pliki

Programowanie obiektowe (C#) - WYKŁAD - (K. Molenda)

/ po-2021-22L-wyklad-kmolenda / Repetytorium / Zadanie: Klasa Person, klasa Child, dziedziczenie

W tym zadaniu musisz zaimplementować dziedziczenie dla klas Person i Child. Dla klas wykorzystaj konstruktorów bezparametrycznych i z parametrem rodzica.

Przedstawię Ci kilka wskazówek, które pomogą Ci w implementacji:

- Klasa Person:** Mała zmiana w konstrukcji bezparametrycznej. W konstruktorze dodaj parametr rodzica (możesz go nazwać np. parent). W metodzie konstruktorowej ustaw wartości dla pola FirstName, LastName i Age.
- Klasa Child:** Dziedziczy po klasie Person. W konstruktorze bezparametrycznym ustaw wartości dla FirstName, LastName i Age. W konstruktorze z parametrem rodzica (parent) ustaw wartości dla FirstName, LastName i Age na te same, co ustawione dla rodzica (parent).
- Metoda ToString:** Dla klas Person i Child powiniene zwracać identyczny ciąg znaków, który jest sumą imienia i nazwiska (separowanej spacją), skąd mające się składać wyłącznie z liter, zaczynając się od dużej litery, zaś pozostałe są literami małymi. Stosowne korekty wprowadź w chwili przypisywania wartości (usuń spacje, skoryguj użycie liter malej). Jeśli napis mały zawierać znaki, które nie są literami małymi, lub pusty zapis zgłoś wyjątek typu `ArgumentException` z komunikatem "Wrong name!".
- Metoda ModifyAge:** Dla klas Person i Child. W metodzie konstruktorowej ustaw wartość dla pola Age. W metodzie `ModifyAge` zmień wartość pola `Age` o podany argument. Wszystkie zmiany powinny być dokonane w klasie dziedziczącej (Child).
- Metoda GetAge:** Dla klas Person i Child. W metodzie konstruktorowej ustaw wartość dla pola Age. W metodzie `GetAge` zwracaj wartość pola `Age`.
- Metoda GetParent:** Dla klas Person i Child. W metodzie konstruktorowej ustaw wartość dla pola Parent. W metodzie `GetParent` zwracaj wartość pola `Parent`.
- Metoda WriteLine:** Dla klas Person i Child. W metodzie konstruktorowej ustaw wartość dla pola FirstName, LastName i Age. W metodzie `WriteLine` zapisz do konsoli串接姓氏和名字，中间用空格分隔。

Ważne: Wszystkie zmiany powinny być dokonane w klasie dziedziczącej (Child). W klasie Person nie powinno być żadnych zmian.

Przykład:

Test	Wynik
/* Test: poprawne tworzenie obiektu Person, dane poprawne */ try { Person p = new Person(FamilyName: "Holenda", firstName: "Krzysztof", age: 18); Console.WriteLine(p); } catch(Exception e) { Console.WriteLine(e.Message); }	Krzysztof Holenda (18)
/* Test: poprawne tworzenie obiektu Person, błędne imię lub nazwisko, iherzuje doda pozostałe małe */ try { Person p = new Person(FamilyName: "Holenda", firstName: "krzysztof", age: 18); Console.WriteLine(p); } catch(Exception e) { Console.WriteLine(e.Message); }	Krzysztof Holenda (18)
/* Test: poprawne tworzenie obiektu Person, błędne imię lub nazwisko, iherzuje wiek */ try { Person p = new Person(FamilyName: "Holenda", firstName: "Krzysztof", age: -18); Console.WriteLine(p); } catch(Exception e) { Console.WriteLine(e.Message); }	Age must be positive!
/* Test: modyfikacja obiektu */ try { Person p = new Person(FamilyName: "Holenda", firstName: "Krzysztof", age: 18); p.modifyFirstName("Ja n"); p.modifyFamilyName("Kolenda"); p.modifyAge(35); Console.WriteLine(p); } catch(Exception e) { Console.WriteLine(e.Message); }	Jan Kolenda (35)
/* Test: tworzenie obiektu Child */ try { Person o = new Person(FamilyName: "Holenda", firstName: "Krzysztof", age: 30); Person m = new Person(FamilyName: "Holenda", firstName: "Eva", age: 29); Child d = new Child(FamilyName: "Holenda", firstName: "Anna", age: 10, mother: m, father: o); Console.WriteLine(d); } catch(Exception e) { Console.WriteLine(e.Message); }	Anna Holenda (30) mother: Eva Holenda (29) father: Krzysztof Holenda (30)
/* Test: tworzenie obiektu Child brak rodziców */ try { Child d = new Child(FamilyName: "Holenda", firstName: "Anna", age: 14); Console.WriteLine(d); } catch(Exception e) { Console.WriteLine(e.Message); }	Anna Holenda (14) mother: No data father: No data
/* Test: tworzenie obiektu Child brak jednego z rodziców */ try { Person o = new Person(FamilyName: "Holenda", firstName: "Krzysztof", age: 30); Person m = new Person(FamilyName: "Holenda", firstName: "Eva", age: 29); Child d = new Child(FamilyName: "Holenda", firstName: "Anna", age: 14, father: o); Console.WriteLine(d); d = new Child(FamilyName: "Holenda", firstName: "Anna", age: 14, mother: m); Console.WriteLine(d); } catch(Exception e) { Console.WriteLine(e.Message); }	Anna Holenda (14) mother: No data father: Krzysztof Holenda (30) Anna Holenda (14) mother: Eva Holenda (29) father: No data

Odpowiedź: (system kar: 0 %)

Zresetuj odpowiedź

```

internal class Person
{
    private string _firstName;
    private string _lastName;
    private int _age;

    public string FirstName
    {
        get { return _firstName; }
        protected set
        {
            if (value == null)
                throw new ArgumentException("Wrong name!");
            value = value.Trim();
        }
    }

    public string LastName
    {
        get { return _lastName; }
        protected set
        {
            if (value == null)
                throw new ArgumentException("Wrong name!");
            value = value.Trim();
        }
    }

    public int Age
    {
        get { return _age; }
        protected set
        {
            if (value < 0)
                throw new ArgumentException("Age must be positive!");
            _age = value;
        }
    }

    public void ModifyAge(int age)
    {
        _age = age;
    }

    public int GetAge()
    {
        return _age;
    }

    public Person GetParent()
    {
        return null;
    }

    public void WriteLine()
    {
        Console.WriteLine($"{_firstName} {_lastName} ({_age})");
    }
}

```

Nawigacja w temacie

Petek Krystian

Zapisz podejście ..

```

        wyjscie;
    }
    if (i == 0)
        wyjscie += char.ToUpper(value[i]);
    else
        wyjscie += char.ToLower(value[i]);
)
for (int i = 0; i < wyjscie.Length; i++)
{
    if (!char.IsLetter(wyjscie[i]))
        throw new ArgumentException("Wrong name!");
}
}
_FirstName = wyjscie;
}
}

public string FamilyName
{
    get { return _lastName; }
    protected set
    {
        if (value == null)
            throw new ArgumentException("Wrong name!");
        value = value.Trim();
        string wyjscie = "";
        for (int i = 0; i < value.Length; i++)
        {
            if (value[i] == ' ')
                continue;
            if (i == 0)
                wyjscie += char.ToUpper(value[i]);
            else
                wyjscie += char.ToLower(value[i]);
        }
        for (int i = 0; i < wyjscie.Length; i++)
        {
            if (!char.IsLetter(wyjscie[i]))
                throw new ArgumentException("Wrong name!");
        }
        _LastName = wyjscie;
    }
}

public int Age
{
    get
    {
        return _Age;
    }
    protected set
    {
        if (value < 0)
            throw new ArgumentException("Age must be positive!");
        else
            _Age = value;
    }
}

public Person(string firstName, string familyName, int age)
{
    FirstName = firstName;
    FamilyName = familyName;
    Age = age;
}

```

Sprawdz

Test	Oczekiwane	Otrzymane	
✓ /* Test: poprawne tworzenie obiektu Person, dane poprawne */ try { Person p = new Person(familyName: "Holenda", firstName: "Krzysztof", age: 18); Console.WriteLine(p); } catch(Exception e) { Console.WriteLine(e.Message); }	Krzysztof Holenda (18)	Krzysztof Holenda (18)	✓
✓ /* Test: poprawne tworzenie obiektu Person, błędne imię lub nazwisko, spacje przed i po */ try { Person p = new Person(familyName: " Holenda ", firstName: " Krzysztof ", age: 18); Console.WriteLine(p); } catch(Exception e) { Console.WriteLine(e.Message); }	Krzysztof Holenda (18)	Krzysztof Holenda (18)	✓
✓ /* Test: poprawne tworzenie obiektu Person, błędne imię lub nazwisko, spacje w środku */ try { Person p = new Person(familyName: "Holen da", firstName: "Krzysztof.", age: 18); Console.WriteLine(p); } catch(Exception e) { Console.WriteLine(e.Message); }	Wrong name!	Wrong name!	✓
✓ /* Test: poprawne tworzenie obiektu Person, błędne imię lub nazwisko, nie tylko litery */ try { Person p = new Person(familyName: "Holenda", firstName: "Krzysztof.", age: 18); Console.WriteLine(p); } catch(Exception e) { Console.WriteLine(e.Message); }	Wrong name!	Wrong name!	✓
✓ /* Test: poprawne tworzenie obiektu Person, błędne imię lub nazwisko, piersza duża pozostałe małe */ try { Person p = new Person(familyName: "Holenda", firstName: "krzysztof", age: 18); Console.WriteLine(p); } catch(Exception e) { Console.WriteLine(e.Message); }	Krzysztof Holenda (18)	Krzysztof Holenda (18)	✓
✓ /* Test: poprawne tworzenie obiektu Person, błędne imię lub nazwisko, niewłaściwy wiek */ try { Person p = new Person(familyName: "Holenda", firstName: "krzysztof", age: -18); Console.WriteLine(p); } catch(Exception e) { Console.WriteLine(e.Message); }	Age must be positive!	Age must be positive!	✓
✓ /* Test: poprawne tworzenie obiektu Person, błędne imię lub nazwisko, pusty ciąg */ try { Person p = new Person(familyName: " ", firstName: "Krzysztof", age: 18); Console.WriteLine(p); } catch(Exception e) { Console.WriteLine(e.Message); }	Wrong name!	Wrong name!	✓
✓ /* Test: modyfikacja obiektu */ try { Person p = new Person(familyName: "Holenda", firstName: "Krzysztof", age: 18); p.modifyFirstName(" Jan "); p.modifyFamilyName("holenda"); p.modifyAge(35); Console.WriteLine(p); } catch(Exception e) { Console.WriteLine(e.Message); }	Ian Kolenda (35)	Ian Kolenda (35)	✓
✓ /* Test: tworzenie obiektu Child */ try { Person o = new Person(familyName: "Holenda", firstName: "Krzysztof", age: 30); Person m = new Person(familyName: "Holenda", firstName: "Ewa", age: 29); Child d = new Child(familyName: "Holenda", firstName: "Anna", age: 10, mother: o, father: m); }	Ana Holenda (10) mother: Ewa Holenda (29) father: Krzysztof Holenda (30)	Ana Holenda (10) mother: Ewa Holenda (29) father: Krzysztof Holenda (30)	✓

```csharp         }         catch( Exception e )         {             Console.WriteLine(e.Message);         }     }      /* Test: tworzenie obiektu Child      * zły wiek */     try     {         Person o = new Person(familyName: "Holenda", firstName: "Krzysztof", age: 30);         Person m = new Person(familyName: "Holenda", firstName: "Ewa", age: 29);         Child d = new Child(familyName: "Holenda", firstName: "Anna", age: 10,                             mother: m, father: o);         Console.WriteLine(d);     }     catch( Exception e )     {         Console.WriteLine(e.Message);     } }  /* Test: tworzenie obiektu Child  * zły wiek */ try {     Person o = new Person(familyName: "Holenda", firstName: "Krzysztof", age: 30);     Person m = new Person(familyName: "Holenda", firstName: "Ewa", age: 29);     Child d = new Child(familyName: "Holenda", firstName: "Anna", age: -15,                         mother: m, father: o);     Console.WriteLine(d); } catch( Exception e ) {     Console.WriteLine(e.Message); }  /* Test: tworzenie obiektu Child  * brak rodziców */ try {     Child d = new Child(familyName: "Holenda", firstName: "Anna", age: 14);      Console.WriteLine(d); } catch( Exception e ) {     Console.WriteLine(e.Message); }  /* Test: tworzenie obiektu Child  * brak jednego i rodziców */ try {     Person o = new Person(familyName: "Holenda", firstName: "Krzysztof", age: 30);     Person m = new Person(familyName: "Holenda", firstName: "Ewa", age: 29);     Child d = new Child(familyName: "Holenda", firstName: "Anna", age: 14, father: o);     Console.WriteLine(d);     d = new Child(familyName: "Holenda", firstName: "Anna", age: 14, mother: m);     Console.WriteLine(d); } catch( Exception e ) {     Console.WriteLine(e.Message); }  /* Test: tworzenie obiektu Child  * modyfikacja danych */ try {     Person o = new Person(familyName: "Holenda", firstName: "Krzysztof", age: 30);     Person m = new Person(familyName: "Holenda", firstName: "Ewa", age: 29);     Child d = new Child(familyName: "Holenda", firstName: "Anna", age: 14,                         mother: m, father: o);     d.modifyFirstName("Aneta");     Console.WriteLine(d);     d.modifyFirstName("Kolenda");     Console.WriteLine(d);     d.modifyAge(18);     Console.WriteLine(d); } catch( Exception e ) {     Console.WriteLine(e.Message); } ```		Child's age must be less than 15!	Child's age must be less than 15!	✓
		Age must be positive!	Age must be positive!	✓
		Anna Holenda (14) mother: No data father: No data	Anna Holenda (14) mother: No data father: No data	✓
		Anna Holenda (14) mother: No data father: Krzysztof Holenda (30) Anna Holenda (14) mother: Ewa Holenda (29) father: No data	Anna Holenda (14) mother: No data father: Krzysztof Holenda (30) Anna Holenda (14) mother: Ewa Holenda (29) father: No data	✓
		Ewa Holenda (14) mother: Ewa Holenda (29) father: Krzysztof Holenda (30) Kolenda Holenda (14) mother: Ewa Holenda (29) father: Krzysztof Holenda (30) Child's age must be less than 15!	Ewa Holenda (14) mother: Ewa Holenda (29) father: Krzysztof Holenda (30) Kolenda Holenda (14) mother: Ewa Holenda (29) father: Krzysztof Holenda (30) Child's age must be less than 15!	✓
		Krzysztof Holenda (18)	Krzysztof Holenda (18)	✓

Przeszedł wszystkie testy! ✓

Pogratuluj  
Punkty dla tej odpowiedzi: 5,00/5,00.

◀ Zadanie: Kolekcje. Agregator logów

Przejdź do...

Zadanie: Konto w banku (part 1) ▶

Zapisz podejście ...

