Cel zajęć. Celem zajęć jest zapoznanie z praktycznymi aspektami projektowania oraz implementacji klas abstrakcyjnych i interfejsów.

Wprowadzenie teoretyczne. Rozważana w ramach niniejszych zajęć tematyka jest ważna, gdyż klasy abstrakcyjne i interfejsy są powszechnie wykorzystywane przy okazji stosowania mechanizmu dziedziczenia. Aby ze zrozumieniem zrealizować zadania, przewidziane do wykonania w ramach zajęć laboratoryjnych, należy znać znaczenie pojęć takich jak klasa abstrakcyjna, metoda abstrakcyjna, interfejs.

Klasy i metody abstrakcyjne

Klasa abstrakcyjna to klasa, która nie posiada swoich reprezentantów pod postacią obiektów. Jest ona wykorzystywana wyłącznie w roli klasy bazowej dla innych klas. Klasa potomna względem klasy abstrakcyjnej musi implementować jej wszystkie abstrakcyjne metody i właściwości. Klasa abstrakcyjna może zawierać także pola oraz metody i właściwości, które nie są abstrakcyjne. Klasa, która zawiera abstrakcyjną właściwość lub metodę, również musi być abstrakcyjna. Klasa abstrakcyjna jest oznaczona modyfikatorem „abstract".

Metoda abstrakcyjna jest metodą oznaczoną modyfikatorem „abstract". Posiada ona jedynie deklarację w klasie abstrakcyjnej. Definicja metody znajduje się w klasach potomnych, dziedziczących po klasie abstrakcyjnej. Metody abstrakcyjne nie mogą być prywatne.

Zadanie 1 – klasa abstrakcyjna:

1. Utwórz klasę abstrakcyjną o nazwie **FiguraGeometryczna**.
2. W klasie utwórz pole protected typu **string** o nazwie **nazwa** oraz publiczną właściwość typu **string** o nazwie **Nazwa**. Właściwość **Nazwa** przyjmuje wartość pola **nazwa**. Użyj do tego celu akcesoria **get**.
3. W klasie FiguraGeometryczna utwórz abstrakcyjną metodę **ObliczPole()**, która zwraca wynik typu **double**. Metoda ta oblicza pole powierzchni danej figury geometrycznej.

Zadanie 2 – klasy dziedziczące z klasy abstrakcyjnej:

4. Utwórz klasę dziedziczącą z klasy abstrakcyjnej o nazwie **Kolo**. W klasie tej oprócz metody abstrakcyjnej zaimplementuj prywatne pole o nazwie **promien.**
5. Zaimplementuj także konstruktor klasy **Kolo**, który jako argument przyjmuje zmienną **_promien**. W konstruktorze oprócz ustawienia wartości pola **promien**, ustaw także wartość polu **nazwa**. W tym przypadku będzie to **nazwa="Koło"**.
6. Wypełnij także metodę **ObliczPole()** w taki sposób aby obliczała pole koła.
7. Utwórz kolejne klasy dziedziczące z klasy abstrakcyjnej dla figur **Kwadrat**, **Prostokat**, **Trojkat**.
8. Dla każdej z nowo utworzonych klas **Kwadrat**, **Prostokat**, **Trojkat** dodaj odpowiednie pola, potrzebne do obliczenia pola powierzchni danej figury.
9. Dla każdej z klas utwórz odpowiedni konstruktor oraz zaimplementuj metodę **ObliczPole()**.

Zadanie 3 – tworzenie obiektów klas w pętli głównej programu:

10. Utwórz po jednym obiekcie każdej klasy.
11. Wartości boków i innych długości potrzebnych do obliczenia pola powierzchni figury ustaw według własnego uznania.
12. Wyświetl wartości obliczonych pól powierzchni w poniższy sposób:

        Nazwa – pole powierzchni: X.X

    Jako nazwę figury proszę wyświetlić wartość pola **Nazwa**.

Zadanie 4 – tworzenie interfejsu i klas:

13. Utwórz interfejs **FiguraG**, który zawiera właściwość **Nazwa** oraz metodę **ObliczPole()**
14. Utwórz cztery nowe klasy opisujące cztery figury z poprzednich zadań. Klasy te będą dziedziczyły z interfejsu **FiguraG**
15. W nowo utworzonych klasach dodaj odpowiednie pola oraz konstruktory.
16. Utwórz obiekty nowych klas w pętli głównej programu
17. Wartości boków i innych długości potrzebnych do obliczenia pola powierzchni figury ustaw według własnego uznania.
18. Wyświetl wartości obliczonych pól powierzchni w poniższy sposób:

        Nazwa – pole powierzchni: X.X

    Jako nazwę figury proszę wyświetlić wartość pola **Nazwa**.