**Kolokwium 1**

**1)Omów model dziedziczenia w jezyku C#. Zademonstruj na przykładowym kodzie.**

Jednym z najważniejszych pojęć w programowaniu obiektowym jest dziedzicznie. Dziedziczenie pozwala nam na zdefiniowanie klasy w uogólnieniu innej klasy, co pozwala na łatwiejsze tworzenie i zarządzanie aplikacją. Daje również możliwość ponowego wykorzystania kodu i przyśpiesza czas jego implementacji.   
  
Podczas tworzenia nowej klasy, zamiast pisać zupełnie od nowa wszystkie składowe tej klasy, programista może powiedzieć, że nowa klasa ma dziedziczyć z istniejącej już klasy. Istniejąca klasa nazywana jest klasą **bazową** a nowa klasa dziedzicząca po klasie bazowej nosi nazwę klasy **pochodnej**.

modyfikator\_dostepu class klasa\_bazowa

{

    …

}

class klasa\_pochodna : klasa\_bazowa

{

    …

}

 Oto jak dziedziczenie zachodziłoby pomiędzy klasą figury , a klasą prostokąt.

class Figury

{

public string kolorWypelnienia;

public string kolorObramowania;

public int wielkoscObramowania;

}

class Prostokąt :Figury

{

}

Teraz klasa Prostokąt będzie posiadała pola jak kolor i obramowanie. Każda figura powinna mieć informacje o tym w jakim kolorze jest wypełniona i obramowana.

Język C# nie obsługuje wielokrotnego dziedziczenia. Klasa może dziedziczyć po jednej klasie bazowej, ale może implementować wiele interfejsów

### 2) Testowanie wartości brzegowych

Rozwinięciem testów z użyciem klas równoważności jest testowanie wartości brzegowych. Wartość brzegowa to wartość znajdująca się wewnątrz, pomiędzy lub tuż przy granicy danej klasy równoważności.

Przykłady:

* rejestracja osoby w przedziale wiekowym 0 – 120,

testowane wartości brzegowe: {-1, 0, 1, 119, 120, 121}

* długość wiadomości od 1 do 50 znaków:

testowane wartości brzegowe: {0, 1, 2, 49, 50, 51}

* napięcie od 0 do 100 V:

testowane wartości brzegowe: {-1, 0, 1, 99, 100, 101}

**3)Jakie są podstawowe operacje zwiazane z wersjonowaniem wykonywane w systemie Git?Gdzie przechowywane jest repozytorium.**

Konfiguracja danych użytkownika.

Ustawiamy globalne wartości dla naszej nazwy oraz dla naszego adresu e-mail. Dane te ułatwiają naszą identyfikacje w logach zmian. Celem ustawienia nazwy wydajemy polecenie:

git config --global user.name "Imię Nazwisko"

Aby ustawić nasz adres e-mail wykonujemy:

git config --global user.email [imienazwisko@example.com](mailto:imienazwisko@example.com)

Użycie paramteru **--global**powoduje, że wartości te będą używane domyślnie w naszym systemie dla wszystkich repozytoriów.

Dodawanie i usuwanie plików w repozytorium.

Po umieszczeniu w folderze rozboczym plików, musimy je jeszcze dodać do naszego repozytorium. Możemy je dodawać plik po pliku:

git add plik1

git add plik2

Lub dodać wszystkie niedodane pliki:

git add .

Aby usunąć plik z repozytorium wydajemy polecenie:

git rm nazwa\_pliku

Wyświetlenie statusu repozytorium w którym aktualnie się znajdujemy.

git status

Commitowanie wprowadzonych zmian.

git commit -am "Wiadomosc"

Wysyłanie zmian do zdalnego repozytorium.

git push origin master

Pobieranie zmian ze zdalnego repozytorium.

git pull