Úkol 11a

1. **Uveďte alespoň 3 nebo 4 příklady užití vzoru Singleton.**

Singleton se využívá například při připojení k databázi. Toto připojení často potřebujeme pouze jediné.

Dále se používá tam kde je třeba vytvořit pouze jeden objekt. Například systém pro restauraci, kdy instance restaurace bude pouze jediná a není třeba vytvářet další restaurace.

Využití při centralizované správě nějakého systému, kdy centrum je pouze jediné nebo sdílení jediného souboru nebo složky více uživateli.

1. **Uveďte alespoň 2 příklady užití vzoru UtilityClass.**UtilityClass je knihovní třída, kterou použijeme při vytváření třídy, ve které máme uložené často používané metody.

Použití je například při vytváření vlastní matematické knihovny s nějakými předdefinovanými metodami.

Další využití může být například při práci s například s finančními doklady, kde v jedné třídě máme uložené metody pro správu dokladů.

1. **Proveďte implementaci vzoru Singleton nad zvoleným problémem. Otestujte funkčnost vzoru.**Pro jednoduchost jsme zvolili objekt s jedním parametrem String. Zajištění jedinečnosti instance objektu jsme docílili statickým parametrem typu *Singleton* instance. Konstruktor je privátní.

private static Singleton instance;

private String text;

private Singleton(String text){

this.text = text;

}

Metoda pro kontrolu jedinečnosti instance a v případě null parametru instance vytvoření instance. A metoda *toString()* pro výpis hodnoty parametru *text*.

public static Singleton getInstance(String text){

if (instance == null){

System.out.println("Byla vytvořena nová instance.");

instance = new Singleton(text);

}else{

System.out.println("Nelze vytvořit novou instanci, protože instance již … “);

}

return instance;

}

@Override

public String toString(){

return "Text uložený v singleton: " + this.text;

}

Otestování vytvoření dvou instancí a výpis jejich hodnot.

public static void main(String[] args) {

System.out.println("Vytvoření první instance s názvem objekt:");

Singleton objekt = Singleton.getInstance("První singleton");

System.out.println(objekt.toString() + "\n");

System.out.println("Vytvoření druhé instance s názvem objekt2");

Singleton objekt2 = Singleton.getInstance("Druhý singleton");

System.out.println(objekt2.toString() + "\n");

}

Výstup programu. Z výstupu programu je vidět, že i když vytvoříme druhý objekt s jinými daty, tak se i přesto vypíše hodnota první instance. Tak jsme zabezpečili vytvoření jediné instance objektu.

