gds2 Gottlieb-Daimler-Schule 2
Technisches Schulzentrum Sindelfingen
mit Abteilung Akademie für Datenverarbeitung

1 Ausnahmebehandlung (Exception Handling)

Ziel: Exceptions von Standardklassen behandeln (try-catch) und Standard Exceptions auslösen (throw)

1.1. Kopieren Sie sich von Moodle die Zip-Datei „HexDezimalHex“ und fügen Sie die gegebenen Klassen einem neue Package hinzu.
Fügen Sie in `class KonvertiereUI` bei `case 1` eine Ausnahmebehandlung mit „try-catch“ ein, so dass bei Falscheingaben eine Fehlermeldung erscheint und das Programm ohne „Absturz“ wieder das Menü anzeigt.

1.2. Kopieren Sie im Projekt Explorer von Eclipse die Datei `KonvertiereUI.java` und fügen Sie die Kopie im gleichen Package mit dem Namen `KonvertiereUI_v2.java` wieder ein.
Ändern Sie die Fehlerbehandlung in `case 1` jetzt so ab, dass der Benutzer nach einer Falscheingabe direkt die Möglichkeit zur erneuten Eingabe bekommt, ohne dass voher nochmals das Menü angezeigt wird.

```
public static void main(String[] args) throws IOException
{
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    int zahl = 0;
    String hexzahl;
    int wahl;
    boolean bFehler = false;

    do
    {
        System.out.println("\n(1) Dezimal --> Hexadezimal umrechnen");
        System.out.println("(2) Hexadezimal --> Dezimal umrechnen");
        System.out.println("(3) Programm beenden");
        System.out.println("-------------------------------------");
        System.out.print("Ihre Wahl: ");
        wahl = Integer.parseInt((br.readLine()));

        switch (wahl)
        {
        case 1:
            do
            {
                bFehler = false;
                System.out.print("Geben Sie eine Dezimalzahl ein: ");
                try
                {
                    zahl = Integer.parseInt((br.readLine()));
                    System.out.printf("Das entspricht Hexadezimal = %X\n", zahl);
                }
                catch (NumberFormatException nfe)
                {
                    System.out.println("Nur Dezimalzahlen erlaubt!");
                    bFehler = true;
                }
            } while (bFehler);

            break;
        case 2:
            System.out.print("Geben Sie eine Hexzahl ein: ");
            hexzahl = br.readLine();
            zahl = LeseMethoden.hexToInt32(hexzahl);
            System.out.printf("Das entspricht dezimal = %d\n", zahl);
            break;
        default:
            System.out.println("Programm beendet");
            break;
        }
    } while (wahl != 3);
```

1.3. Erstellen Sie nochmals eine neue Kopie unter dem Namen `KonvertiereUI_v3.java`. Jetzt soll das Einlesen inklusive Fehlerabhandlung mit erneuter Eingabemöglichkeit für den Benutzer in eine Klassenmethode der Klasse `LeseMethoden` ausgelagert werden. Diese Klassenmethode hat folgende Signatur:

```
public static int ReadInt32(String eingabeaufforderung)
```

Der Programmtext bei case 1 reduziert sich dann auf folgendes:

```
case 1:
    zahl = LeseMethoden.ReadInt32("Bitte eine Dezimalzahl eingeben: ");
    System.out.printf("Das entspricht Hexadezimal = %X\n", zahl);
break;
```

```
public static int ReadInt32(String eingabeaufforderung) throws IOException
{
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

    boolean gueltig;
    String input;
    int ergebnis = 0;
    do
    {
        System.out.print(eingabeaufforderung);
        try
        {
            input = (br.readLine());
            ergebnis = Integer.parseInt(input);
            gueltig = true;
        }
        catch (NumberFormatException formatausnahme)
        {
            System.out.println("Fehler: Bitte nur ganze Zahlen eingeben");
            //System.out.println(formatausnahme.getMessage());
            gueltig = false;
        }

    } while (!gueltig);

    return ergebnis;
}
```

1.4. Die in 1.3 erstellte Methode kann jetzt auch für das Einlesen der Menüauswahl verwendet werden, um hier ebenfalls einen Programmabsturz bei Fehleingabe zu verhindern.

1.5. Bei `case 2` ist es im Moment noch so, dass bei Falscheingaben durch die Benutzer das Programm eine Fehlermeldung ausgibt, sich aber danach beendet.

Probieren Sie dies selbst einmal aus!

Um dies zu verhindern soll eine weitere Klassenmethode der Klasse `LeseMethoden` hinzugefügt werden. Diese Klassenmethode hat folgende Signatur:

```
public static int ReadInt32Hex(String eingabeaufforderung)
```

Diese Methode stimmt (abgesehen vom Namen) fast vollständig mit der Methode von 1.3

überein. Der Unterschied besteht in der Konvertierung des Eingabestrings. Hierzu wird statt `parseInt`() die Methode `hexToInt32`() verwendet, die in der Klasse `LeseMethoden` bereits vorhanden ist.

Wenn Sie dies Methode implementiert haben können Sie `case 2` entsprechend `case 1` abändern.

```
public static void main(String[] args) throws IOException
{
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    int zahl = 0;
    int wahl;

    do
    {
        System.out.println("\n(1) Dezimal --> Hexadezimal umrechnen");
        System.out.println("(2) Hexadezimal --> Dezimal umrechnen");
        System.out.println("(3) Programm beenden");
        System.out.println("---------------------------------------");
        wahl = LeseMethoden_v3.ReadInt32("Ihre Wahl: ");   // Aufgabe 1.4

        switch (wahl)
        {
        case 1:   // Aufgabe 1.3
            zahl = LeseMethoden_v3.ReadInt32("Bitte eine Dezimalzahl eingeben: ");
            System.out.printf("Das entspricht Hexadezimal = %X\n", zahl);
            break;
        case 2:   // Aufgabe 1.5
            zahl = LeseMethoden_v3.ReadInt32Hex("Bitte eine Hexzahl eingeben: ");
            System.out.printf("Das entspricht dezimal = %d\n", zahl);
            break;
        default:
            System.out.println("Programm beendet");
            break;
        }
    } while (wahl != 3);

}
```

```
public static int ReadInt32Hex(String eingabeaufforderung) throws IOException
{
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

    boolean gueltig;
    String input;
    int ergebnis = 0;
    do
    {
        System.out.print(eingabeaufforderung);
        try
        {
            input = (br.readLine());
            ergebnis = hexToInt32(input);
            //ergebnis = Integer.parseInt(input, 16);
            gueltig = true;
        }
        catch (NumberFormatException formatausnahme)
        {
            System.out.println("Fehler: Bitte nur Hexzahlen eingeben");
            //System.out.println(formatausnahme.getMessage());
            gueltig = false;
        }

    } while (!gueltig);

    return ergebnis;
}
```

1.6. Wenn man jetzt das Programm testet, so hat sich das Verhalten bei falsch eingegebenen Hexzahlen noch nicht verändert.

Die Methode `hexToInt32`() ruft die Methode `hexDigitToInt`() auf.

In dieser Methode muss der Befehl `System.exit()` (der das Programm beendet) ersetzt werden durch einen `throw` Befehl, der eine `NumberFormatException` „wirft“.

Wenn man nach dieser Änderung wieder testet, verhält sich das Programm wie erwartet.

```
static private int hexDigitToInt(char digit)
{
    int wert = 0;

    if (digit >= '0' && digit <= '9')
        wert = digit - '0';
    else
        if (digit >= 'a' && digit <= 'f' || digit >= 'A' && digit <= 'F')
    {
        wert = 10 + Character.toLowerCase(digit) - 'a';
    }
    else
    {
        //System.out.println("Nur Hexziffern erlaubt");
        //System.exit(0);
        throw (new NumberFormatException("Nur Hexziffern erlaubt"));
    }

    return wert;
}
```

1.7. Analysieren Sie die Methoden `hexToInt32()` und `hexDigitToInt()` und versuchen Sie nachzuvollziehen, wie der Eingabe-String mit Hexziffern in das interne Integer-Format umgewandelt wird.

1.8. Ersetzen Sie in der Methode `ReadInt32Hex()` den Aufruf von `hexToInt32()` durch `parseInt(input, 16)` . Die Methode haben wir bereits verwendet, allerdings mit nur einem Parameter zur Konvertierung von Dezimalzahl Strings. Die Bedeutung des 2. Parameters im vorliegenden Fall dürfe leicht zu erraten sein.