Лабораторная работа №1

ИСКЛЮЧЕНИЯ

Исключения – это ошибки, возникающие в ходе выполнения программы. Распространенными примерами таких ошибок являются: деление на ноль, ошибка перевода числа в строку, вызов метода у null-объекта, выход за границы массива и т.д. В случае возникновения такой ошибки, выполнение метода прерывается, создается и выбрасывается объект специального типа, соответствующего возникшей ошибке.

Примеры:

```
SomeClass c = null;
c.someMethod(); // возникает объект-исключение типа NullPointerException

int i = Integer.parseInt("abc123"); //Возникает объект-исключение типа NumberFormatException

int a[] = new int[5];
a[10] = 100; //Возникает объект-исключение типа ArrayIndexOutOfBoundsException
```

Описание всех типов исключений можно найти в документации по JAVA. Общим для всех типов исключений является то, что все они наследуются от класса `Exception`.

Исключения могут быть специальным образом обработаны (перехвачены). Если исключение не было обработано, то оно передается в метод, вызвавший метод в котором возникло данное исключение. Если исключение не обрабатывается и в этом методе, то оно снова передается на уровень выше и так далее, пока не дойдет до метода `main()`. Если в `main()` исключение опять же не обрабатывается, происходит завершение работы программы.

Обработка исключений

Для обработки исключений блок кода, в котором может произойти ошибка, заключается внутрь специальной конструкции `try {…}`, после которой идет один или несколько блоков `catch(<тип исключения> переменная) {…}`, которые содержат операции, выполняемые в случае возникновения исключения указанного типа. Сам объект исключения передается по ссылке в переменную, указанную в блоке `catch`.

Пример:

```
public void someMethod(String s) {
  try {
         ...
         s = s.trim();  // может возникнуть исключение типа NullPointerException

         int i = Integer.parseInt(s); // может возникнуть NumberFormatException

         …
  }
  // выполняется в случае возникновения NullPointerException
  catch (NullPointerException ex) {
         System.out.println("Ошибка: s равняется null");
  }
  // выполняется в случае возникновения NumberFormatException
  catch (NumberFormatException ex) {
         System.out.println("Ошибка: некорректное значение s ");
  }
  // выполняется в случае возникновения любого другого исключения
  catch (Exception ex) {
         System.out.println("Неизвестная ошибка");
  }

}
```

В приведенном примере, при вызове метода `someMethod` начинается выполнение содержимого блока `try`. Если, например, переменная `s` равна null, при выполнении операции `s.trim()` формируется объект-исключение типа `NullPointerException`, работа блока `try` прерывается и происходит переход к блокам `catch`. Далее, начинается последовательный поиск и выполнение блока `catch`, в котором указан тип исключения `NullPointerException`. В вышеописанной ситуации произойдет вывод на экран сообщения «Ошибка: s равняется null» и на этом выполнение метода завершается.

Кроме того, в приведенном примере присутствует блок `catch (Exception …)` Так как тип `Exception` является базовым для всех типов исключений, и, следовательно, любое исключение может быть приведено к данному типу, блок `catch` с указанием типа `Exception`, будет отлавливать все исключения вне зависимости от типа (если оно не было отловлено ранее). Блок `catch(Exception …)` является необязательным, но в случае его наличия, он должен следовать после других блоков `catch`.

Класс `Exception` (и, соответственно, все его наследники) содержит несколько полезных методов, которые могут быть использованы в блоке catch для вывода информации о возникшей ошибке:

`String getMessage()` – возвращает строку-описание возникшего исключения.

`printStackTrace()` – выводит в консоль последовательность (стек) вызовов методов, в результате которого произошла данная ошибка.

Пример:

```
try {
 ...
}
catch (NullPointerException ex) {
       System.out.println("Ошибка: " + ex.getMessage());
       ex.printStackTrace();
}
```

Создание и выброс собственных исключений

В некоторых случаях, не предусмотренных в , определенные ситуации в программе необходимо рассматривать как ошибочные и производить самостоятельный выброс исключения. Так, например, можно рассматривать как ошибочную ситуацию, когда при попытке сформировать описание растения значение свойства название равно `null` или пустой строке. Для выброса собственного исключения необходимо создать объект-исключение соответствующего типа и выбросить его с помощью ключевого слова `throw`. Обычно собственные исключения выбрасываются в зависимости от определенного условия.

Пример:

```
public String сформироватьОписание() {
   try {
        ...

        // в случае выполнения условия выбрасывается собственное исключение типа Exception, в
        // конструкторе указывается строковое сообщение об ошибке
        if (nazvanie = = null || nazvanie.equals("")) throw new Exception ("Не указано название дерева");

        ...

   } catch (Exception ex) {
        System.out.println(ex.getMessage());
        return null; // в случае ошибки метод возвращает null
   }
}
```

В приведенном примере в блоке `catch` обрабатываются все исключения, включая наше собственное исключение. Если требуется отделить обработку нашего исключения от обработки прочих исключений, следует описать собственный класс исключений, сделав его наследником класса `Exception` (или класса – наследника от `Exception`) и использовать созданный класс при выбросе и обработке исключений.

Пример:

```
// Собственный класс исключения
class MyException extends Exception {

        // Переопределяем конструктор для сохранения возможности передачи строки сообщения об ошибке
        public MyException(String message) {
            super(message);
        }
}
```

Используем созданный класс исключения в методе формирования описания дерева:

```
public String сформироватьОписание() {
   try {
        ...

        // в случае выполнения условия выбрасывается собственное исключение типа MyException, в
        // конструкторе указывается строковое сообщение об ошибке
        if (nazvanie = = null || nazvanie.equals("")) throw new MyException ("Не указано название дерева");

        ...

   } catch (MyException ex) {
        System.out.println(ex.getMessage());
        return null;
   }
}
```

Описание исключений в заголовках методов

В JAVA имеется возможность указывать в заголовке методов типы исключений, которые этот метод может выбрасывать. Это нужно для того, чтобы предупредить программистов, использующих этот метод, об исключительных ситуациях, которые могут возникнуть при вызове данного метода и заставить обработать эти исключения.

Для указания типов выбрасываемых исключений в заголовке метода необходимо после списка параметров указать ключевое слово throws и затем перечислить типы исключений через запятую.

Пример:

```
public void someMethod() throws MyException, NullPointerException, Exception {

 ...

}
```

Описанный подход дает возможность делегировать обязанность обработки исключений на вызывающий метод, так как в методе, возбуждающем исключение, можно его не обрабатывать.

```
public String сформироватьОписание() throws MyException {
        ...

        // в случае выполнения условия выбрасывается собственное исключение типа MyException, в
        // конструкторе указывается строковое сообщение об ошибке
        if (nazvanie = = null || nazvanie.equals("")) throw new MyException ("Не указано название дерева");

        ...

}
```

Повторный выброс исключения

Выше были рассмотрены способы обработки исключений и передача исключений в вызывающий метод. Однако имеется возможность совмещения этих подходов, т.е. исключение может быть обработано и выброшено в вызывающий метод:

```
public String сформироватьОписание() throws MyException {
   try {
        ...

        if (nazvanie = = null || nazvanie.equals("")) throw new MyException ("Не указано название дерева");

        ...

   } catch (MyException ex) {
        System.out.println(ex.getMessage());
        throw ex; // повторный выброс исключения (операция return отсутствует)
   }
}
```

ВАРИАНТЫ ЗАДАНИЙ

Вариант № 1

Смоделировать структуру предприятия:

Классы	Свойства и методы
Фирма	название (get, set)
Отдел	название (get, set) количество сотрудников (get, set)
Сотрудник	фио (get, set) должность (get, set) оклад (get, set) рассчитать зарплату()
Штатный сотрудник	премия (get, set) рассчитать зарплату()
Сотрудник по контракту	рассчитать зарплату()

а) Обработать все исключения с помощью блока try…catch(Exception …) в методе «рассчитать зарплату» классов Штатный сотрудник и Сотрудник по контракту. При возникновении исключения выводить на экран сообщение об ошибке и стек операций.

Описать собственный класс исключений PremiyaException. В методе «рассчитать зарплату» класса Штатный сотрудник выбрасывать собственное исключение типа PremiyaException при отрицательном значении свойства Премия. В этом же методе обработать PremiyaException в блоке catch. При возникновении исключения выводить сообщение об ошибке на экран.

В основной программе проверить работу блока обработки исключений метода «рассчитать зарплату».

б) Описать собственный класс исключений OkladException. Описать конструктор класса Сотрудник как выбрасывающий исключение (throws OkladException). В конструкторе реализовать проверку значения оклада, при отрицательном значении выбрасывать собственное исключение типа OkladException. Обработать исключения OkladException с помощью блока try…catch, в блоке обработки исключений вывести на экран сообщение «Невозможно создать сотрудника – указан отрицательный оклад: <оклад> » и повторно выбросить исключение.

В основной программе (main) обработать вызов конструктора класса Сотрудник и проверить работу обработчика исключения.

Вариант № 2

Смоделировать структуру банка:

Классы	Свойства
Банк	название (get, set)
Филиал	название (get, set) общая сумма вкладов (get, set)
Вклад	фио вкладчика (get, set) сумма вклада (get, set) рассчитать сумму вклада (количество месяцев)
Долгосрочный вклад	рассчитать сумму вклада (количество месяцев)
Вклад до востребования	рассчитать сумму вклада (количество месяцев)

а) Обработать все исключения с помощью блока try…catch(Exception …) в методе «рассчитать сумму вклада (количество месяцев)» классов Долгосрочный вклад и Вклад до востребования. При возникновении исключения выводить на экран сообщение об ошибке и стек операций.

Описать собственный класс исключений KolichestvoException. В методе «рассчитать сумму вклада (количество месяцев)» классов Долгосрочный вклад и Вклад до востребования осуществить проверку значения параметра Количество месяцев - выбрасывать собственное исключение типа KolichestvoException при отрицательном значении. В этом же методе обработать KolichestvoException в блоке catch. При возникновении исключения выводить сообщение об ошибке на экран.

В основной программе проверить работу блока обработки исключений метода «рассчитать сумму вклада (количество месяцев)».

б) Описать собственный класс исключений VkladException. Описать конструктор класса Вклад как выбрасывающий исключение (throws VkladException). В конструкторе реализовать проверку значения суммы вклада, при отрицательном значении выбрасывать собственное исключение типа VkladException. Обработать исключения VkladException с помощью блока try…catch, в блоке обработки исключений вывести на экран сообщение «Невозможно создать вклад – указана отрицательная сумма вклада: <сумма вклада> » и повторно выбросить исключение.

В основной программе (main) обработать вызов конструктора класса Вклад и проверить работу обработчика исключения.

Вариант № 3

Смоделировать структуру аэропорта:

Классы	Свойства
Аэропорт	название (get, set)
Летательный аппарат	название (get, set) макс. количество пассажиров (get, set) взлет()
Самолет	длина полосы разгона (get, set) взлет()
Вертолет	взлет()
Пассажир	фио (get, set) №посадочного места (get, set)

а) Обработать все исключения с помощью блока try…catch(Exception …) в методе «взлет» классов Самолет Вертолет. При возникновении исключения выводить на экран сообщение об ошибке и стек операций.

Описать собственный класс исключений KolichestvoException. В методе «добавить пассажира» класса Летательный аппарат выбрасывать собственное исключение типа KolichestvoException при превышении максимального количества пассажиров (сравнивать текущее количество пассажиров и значение свойства «макс. количество пассажиров»). В этом же методе обработать KolichestvoException в блоке catch. При возникновении исключения выводить сообщение об ошибке на экран.

В основной программе (main) проверить работу блока обработки исключений методов «взлет» и «добавить пассажира».

б) Описать собственный класс исключений PolosaRazgonaException. Описать конструктор класса Самолет как выбрасывающий исключение (throws PolosaRazgonaException). В конструкторе реализовать проверку значения длины полосы разгона, при отрицательном значении или нулевом значении выбрасывать собственное исключение типа PolosaRazgonaException. Обработать исключения PolosaRazgonaException с помощью блока try…catch, в блоке обработки исключений вывести на экран сообщение «Невозможно создать самолет – указана некорректная длина полосы разгона: <длина полосы разгона> » и повторно выбросить исключение.

В основной программе (main) обработать вызов конструктора класса Самолет и проверить работу обработчика исключения.

Вариант № 4

Смоделировать структуру библиотеки:

Классы	Свойства
Библиотека	название (get, set)
Отдел (по жанрам)	название жанра (get, set) количество изданий (get, set)
Издание	название (get, set) автор (get, set) год издания (get, set) сформировать описание()
Книга	резюме (get, set) сформировать описание()
Журнал	статьи (get, add) сформировать описание()

а) Обработать все исключения с помощью блока try…catch(Exception …) в методе «сформировать описание» классов Книга и Журнал. При возникновении исключения выводить на экран сообщение об ошибке и стек операций.

Описать собственный класс исключений BookException. В методе «сформировать описание» классов Книга и Журнал осуществить проверку значений свойств Название и Автор – выбрасывать собственное исключение типа BookException при значении null или пустой строке (“”).. В этом же методе обработать BookException в блоке catch. При возникновении исключения выводить сообщение об ошибке на экран.

В основной программе (main) проверить работу блока обработки исключений метода «сформировать описание».

б) Описать собственный класс исключений GodIzdaniyaException. Описать конструктор класса Издание как выбрасывающий исключение (throws GodIzdaniyaException). В конструкторе реализовать проверку значения года издания, при отрицательном или нулевом значении выбрасывать собственное исключение типа GodIzdaniyaException. Обработать исключения GodIzdaniyaException с помощью блока try…catch, в блоке обработки исключений вывести на экран сообщение «Невозможно создать издание – указан некорректный год издания: <год издания>» и повторно выбросить исключение.

В основной программе (main) обработать вызов конструктора класса Издания и проверить работу обработчика исключения.

Вариант № 5

Смоделировать структуру компании сотовой связи:

Классы	Свойства
Компания	Название (get, set)
Тариф	название (get, set) количество абонентов (get, set) рассчитать стоимость разговора (количество секунд)
Посекундный тариф	стоимость секунды разговора» (get, set) рассчитать стоимость разговора (количество секунд)
Поминутный тариф	стоимость минуты разговора» (get, set) рассчитать стоимость разговора (количество секунд)
Абонент	фио (get, set) номер телефона (get, set) остаток на счете (get, set)

а) Обработать все исключения с помощью блока try…catch(Exception …) в методе «рассчитать стоимость разговора (количество секунд)» классов Посекундный тариф и Поминутный тариф. При возникновении исключения выводить на экран сообщение об ошибке и стек операций.

Описать собственный класс исключений KolichestvoSekundException. В методе «рассчитать стоимость разговора (количество секунд)» классов Посекундный тариф и Поминутный тариф осуществить проверку значения параметра Количество секунд – выбрасывать собственное исключение типа KolichestvoSekundException при отрицательном значении. В этом же методе обработать KolichestvoSekundException в блоке catch. При возникновении исключения выводить сообщение об ошибке на экран.

В основной программе (main) проверить работу блока обработки исключений метода «рассчитать стоимость разговора (количество секунд)».

б) Описать собственный класс исключений NazvanieException. Описать конструктор класса Тариф как выбрасывающий исключение (throws NazvanieException). В конструкторе реализовать проверку значения свойства Название, при значении null или пустой строке выбрасывать собственное исключение типа NazvanieException. Обработать исключение NazvanieException с помощью блока try…catch, в блоке обработки исключений вывести на экран сообщение «Невозможно создать тариф – не указано название» и повторно выбросить исключение.

В основной программе (main) обработать вызов конструктора класса Тариф и проверить работу обработчика исключения.

Вариант № 6

Смоделировать структуру автосалона:

Классы	Свойства
Автосалон	название (get, set)
Автомобиль	марка (get, set) макс. количество пассажиров (get, set) стоимость (get, set) количество на складе(get, set) `boolean` наличие (get, set)
Заявка на покупку	фио покупателя (get, set) номер телефона (get, set) автомобили (add, remove, get) рассчитать стоимость заказа()
Заявка на приобретение со стенда	рассчитать стоимость заказа()
Заявка на отложенную поставку	процент скидки (get, set) рассчитать стоимость заказа()

а) Обработать все исключения с помощью блока try…catch(Exception …) в методе «рассчитать стоимость заказа» классов Заявка на приобретение со стенда и Заявка на отложенную поставку. При возникновении исключения выводить на экран сообщение об ошибке и стек операций.

Описать собственный класс исключений KolichestvoAvtomobileyException. В методе «рассчитать стоимость заказа» классов Заявка на приобретение со стенда и Заявка на отложенную поставку выбрасывать собственное исключение типа KolichestvoAvtomobileyException, если количество автомобилей в заявке равно нулю. В этом же методе обработать KolichestvoAvtomobileyException в блоке catch. При возникновении исключения выводить сообщение об ошибке на экран.

В основной программе (main) проверить работу блока обработки исключений метода «рассчитать стоимость заказа».

б) Описать собственный класс исключений FIOException. Описать конструктор класса Заявка на покупку как выбрасывающий исключение (throws FIOException). В конструкторе реализовать проверку значения свойства ФИО покупателя, при значении null или пустой строке выбрасывать собственное исключение типа FIOException. Обработать исключение FIOException с помощью блока try…catch, в блоке обработки исключений вывести на экран сообщение «Невозможно создать заявку – не указано ФИО покупателя» и повторно выбросить исключение.

В основной программе (main) обработать вызов конструктора класса Заявка на покупку и проверить работу обработчика исключения.

Вариант № 7

Смоделировать структуру музыкальной коллекции:

Классы	Свойства
Коллекция	название (get, set) фио владельца (get, set)
Музыкальный носитель (альбом)	автор/группа (get, set) жанр (get, set) год выпуска (get, set) общая продолжительность звучания (get, set)
Музыкальное произведение	название (get, set) продолжительность (get, set) сформировать описание()
Песня	текст (get, set) автор текста (get, set) сформировать описание()
Инструментальное произведение	инструменты (get, set) сформировать описание()

а) Обработать все исключения с помощью блока try…catch(Exception …) в методе «сформировать описание» классов Песня и Инструментальное произведение. При возникновении исключения выводить на экран сообщение об ошибке и стек операций.

Описать собственный класс исключений ProdolzhitelnostException. В методе «сформировать описание» классов Песня и Инструментальное произведение осуществить проверку свойства Продолжительность и выбрасывать собственное исключение типа ProdolzhitelnostException при значении свойства меньшем или равном нулю. В этом же методе обработать ProdolzhitelnostException в блоке catch. При возникновении исключения выводить сообщение об ошибке на экран.

В основной программе (main) проверить работу блока обработки исключений метода «сформировать описание».

б) Описать собственный класс исключений NazvanieException. Описать конструктор класса Музыкальное произведение как выбрасывающий исключение (throws NazvanieException). В конструкторе реализовать проверку значения свойства Название, при значении null или пустой строке выбрасывать собственное исключение типа NazvanieException. Обработать исключение NazvanieException с помощью блока try…catch, в блоке обработки исключений вывести на экран сообщение «Невозможно создать музыкальное произведение – не указано название» и повторно выбросить исключение.

В основной программе (main) обработать вызов конструктора класса Музыкальное произведение и проверить работу обработчика исключения.

Вариант № 8

Смоделировать структуру реестра городского жилья:

Классы	Свойства
Город	название (get, set)
Здание	название улицы (get, set) номер дома (get, set) общая площадь (get) базовая ежемесячная оплата за кв.м площади (get, set)
Помещение	номер (get, set) площадь (get, set) рассчитать ежемес. оплату()
Квартира	ФИО жильцов (get, add, remove) рассчитать ежемес. оплату()
Офис	название фирмы-владельца (get, set) вид деятельности (get, set) рассчитать ежемес. оплату()

а) Обработать все исключения с помощью блока try…catch(Exception …) в методе «рассчитать ежемес. оплату» классов Квартира и Офис. При возникновении исключения выводить на экран сообщение об ошибке и стек операций.

Описать собственный класс исключений KolichestvoException. В методе «рассчитать ежемес. оплату» класса Квартира осуществить проверку свойства «ФИО жильцов» и выбрасывать собственное исключение типа KolichestvoException при отсутствии жильцов в квартире (кол-во эл-тов коллекции равно нулю). В этом же методе обработать KolichestvoException в блоке catch. При возникновении исключения выводить сообщение об ошибке на экран.

В основной программе (main) проверить работу блока обработки исключений метода «рассчитать ежемес. оплату».

б) Описать собственный класс исключений PloschadException. Описать конструктор класса Помещение как выбрасывающий исключение (throws PloschadException). В конструкторе реализовать проверку значения свойства Площадь, при значении 0 или отрицательном значении выбрасывать собственное исключение типа PloschadException. Обработать исключение PloschadException с помощью блока try…catch, в блоке обработки исключений вывести на экран сообщение «Невозможно создать помещение – указано некорректное значение площади: <площадь>» и повторно выбросить исключение.

В основной программе (main) обработать вызов конструктора класса Помещение и проверить работу обработчика исключения.

Вариант № 9

Смоделировать структуру зоопарка:

Классы	Свойства
Зоопарк	название (get, set)
Вольер/клетка	номер (get, set) размер (get, set) макс. количество животных (get, set) текущее количество животных (get, set)
Животное	название (get, set) `boolean` хищник (get, set) сформировать описание()
Рыба	глубоководная (get, set) сформировать описание()
Птица	скорость полета (get, set) сформировать описание()
Зверь	среда обитания (get, set) сформировать описание()

а) Обработать все исключения с помощью блока try…catch(Exception …) в методе «сформировать описание» классов Рыба, Птица и Зверь. При возникновении исключения выводить на экран сообщение об ошибке и стек операций.

Описать собственный класс исключений HischnikException. В методе «добавить животное» в классе Клетка при попытке добавить хищника в клетку, где уже содержатся не хищные животные выбрасывать собственное исключение типа HischnikException. В этом же методе обработать HischnikException в блоке catch. При возникновении исключения выводить сообщение об ошибке на экран.

В основной программе (main) проверить работу блока обработки исключений методов «сформировать описание» классов Рыба, Птица и Зверь и метода «добавить животное» класса Клетка.

б) Описать собственный класс исключений NazvanieException. Описать конструктор класса Животное как выбрасывающий исключение (throws NazvanieException). В конструкторе реализовать проверку значения свойства Название, при значении null или пустой строке выбрасывать собственное исключение типа NazvanieException. Обработать исключение NazvanieException с помощью блока try…catch, в блоке обработки исключений вывести на экран сообщение «Невозможно создать животное – не указано название» и повторно выбросить исключение.

В основной программе (main) обработать вызов конструктора класса Животное и проверить работу обработчика исключения.

Вариант № 10

Смоделировать структуру автоматизированного банкомата:

Классы	Свойства
Банк	название (get, set)
Счет	номер (get, set) PIN-код (get, set) остаток (get, set) снять со счета (сумма)
Обычный счет	снять со счета (сумма)
Льготный счет	снять со счета (сумма)
Банкомат	идентификационный номер (get, set) адрес (get, set)

а) Обработать все исключения с помощью блока try…catch(Exception ...) в методе «снять со счета (сумма)» классов Обычный счет и Льготный счет. При возникновении исключения выводить на экран сообщение об ошибке и стек операций.

Описать собственный класс исключений SnyatSoSchetaException. В методе «снять со счета (сумма)» классов Обычный счет и Льготный счет осуществить проверку значения параметра «сумма» и выбрасывать собственное исключение типа SnyatSoSchetaException при отрицательном значении, а также при попытке снять со счета сумму большую, чем остаток на счете. В этом же методе обработать SnyatSoSchetaException в блоке catch. При возникновении исключения выводить сообщение об ошибке на экран.

В основной программе (main) проверить работу блока обработки исключений метода «снять со счета (сумма)».

б) Описать собственный класс исключений OstatokNaScheteException. Описать конструктор класса Счет как выбрасывающий исключение (throws OstatokNaScheteException). В конструкторе реализовать проверку значения остатка на счете, при отрицательном значении выбрасывать собственное исключение типа OstatokNaScheteException. Обработать исключение OstatokNaScheteException с помощью блока try…catch, в блоке обработки исключений вывести на экран сообщение «Невозможно создать счет – указано некорректное значение остатка на счете: <остаток на счете>» и повторно выбросить исключение.

В основной программе (main) обработать вызов конструктора класса Счет и проверить работу обработчика исключения.