Лабораторная 1

Manifest.MF

Файл манифеста называется MANIFEST.MF и находится в каталоге META-INF в JAR. Это просто список пар ключ-значение, называемый заголовками или атрибутами , сгруппированный в разделы.

Эти заголовки предоставляют метаданные, которые помогают нам описывать аспекты нашего JAR, такие как версии пакетов, какой класс приложения выполнять, путь к классам, материал подписи и многое другое.
Файл манифеста добавляется автоматически всякий раз, когда мы создаём JAR.

Точка входа

Любая программа на Java должна иметь "точку входа", т.е. то место, откуда начинается исполнение программы. В Java подобной точкой входа является специальный метод с именем main().

Оператор continue

Позволяет пропустить оставшуюся часть текущей итерации цикла и немедленно перейти к следующей итерации, пропускает текущий шаг цикла.

Varargs

***Varargs* (*Variable Arguments List*, изменяющийся список аргументов) — это способ создания методов, которые могут принимать произвольное количество аргументов одного типа**

- **String format — выводимое на консоль сообщение, включающее параметры форматирования (дескрипторы)**
- **Object… args — произвольное количество значений форматируемых переменных**

Стек в Java

Стек — это линейная структура данных, которая используется для хранения коллекции объектов.

Она основана на принципе Last-In-First-Out (LIFO). Фреймворк коллекций Java предоставляет множество интерфейсов и классов для хранения коллекции объектов.

- Он заполняется и освобождается по мере вызова и завершения новых методов

- Переменные в стеке существуют до тех пор, пока выполняется метод в котором они были созданы
- Если память стека будет заполнена, Java бросит исключение java.lang.StackOverFlowError
- Доступ к этой области памяти осуществляется быстрее, чем к куче
- Является потокобезопасным, поскольку для каждого потока создается свой отдельный стек

Куча - эта область памяти используется для динамического выделения памяти для объектов и классов JRE во время выполнения. Новые объекты всегда создаются в куче, а ссылки на них хранятся в стеке.

Особенности Java

1. Объектно-ориентированность. Джава – инструмент разработки, имеющий полную ориентацию на объекты. Каждый программный компонент включает в себя данные и методы для обработки.
2. Кроссплатформенность. Это одна из основных особенностей языка. Он может работать на разных операционных системах, а также позволяет писать универсальное ПО.
3. Многопоточность. В программах можно одновременно выполнять несколько задач.
4. Наличие сборщика мусора. Джава автоматически управляет выделением памяти, а также его высвобождением.
5. Строгая типизация. У каждой переменной поддерживается свой собственный тип данных, а операции над ними будут проверяться на соответствие типов.

Средства разработки. JDK и JRE. Компиляция и выполнение программы. JAR-архивы.

1. **javac** - Компилирует исходные тексты (файлы с расширением .java) в байт-код (файлы с расширением .class). В одном файле с расширением .java должен находиться только один public- класс, и имя этого класса (без имени пакета) должно совпадать с именем файла (без расширения).
2. **java** - Интерпретатор байт-кода. Запускает Java-программы (файлы с расширением .class). java [параметры] имя_класса [аргументы]. Программа, которую необходимо выполнить, должна представлять собой класс с именем имя_класса (без расширения .class, но с указанием пакета, которому принадлежит класс) и содержать метод main() с описанием: public static void main(String args[])
3. **javadoc** - Создает документацию в формате HTML для указанных пакетов или файлов

JDK - Java Development Kit(комплект Java разработчика). Это набор программных инструментов, необходимых для разработки Java-приложений. В JDK, помимо JRE,

также содержится ряд инструментов разработки — компиляторы, отладчики, JavaDoc и т.д.

JRE - Java Runtime Environment. Это среда исполнения Java, которая обязательно включает в себя реализацию JVM и библиотеки Java-классов (Стандартную библиотеку Java)

JVM - Java Virtual Machine. Это основная часть платформы Java Runtime Environment (JRE), которая интерпретирует байт-код Java для запуска программ. Состоит из Bytecode intepreter, JIT(технология увеличения производительности программн путём компиляции байт-кода в машинный код или в другой формат во время работы программы), Garbage Collector.

.jar файлы в Java представляют собой архивы, которые содержат в себе байт-код, ресурсы (такие как изображения, тексты и т.д.), а также дополнительные файлы, которые могут быть необходимы для работы приложения. Основное назначение .jar файлов — это упаковка и распространение Java приложений или библиотек, которые могут быть запущены на любой машине, где установлена Java.

Нам понадобятся опции -c для создания архива, -f для указания имени архива, -e для указания имени основного класса, который содержит метод main. В конце перечисляются файлы, которые должны попасть в архив

Примитивные типы данных в Java. Приведение типов.

Язык Java является **языком со строгой типизацией.** Типы данных делятся на **примитивные и ссылочные(массивы, классы и интерфейсы)**. Это 8 примитивных типов данных:

byte, short, int, long, float, double, char, boolean

Приведение типов — это механизм конвертации типов, не предусмотренный стандартным преобразованием.

```
double height = 19,12;

int total_height = (int) height;
```

Преобразование типов — это возможность работать со значением в памяти одного типа, как со значением другого.

```
short a = 255, b = 10;

int cur_value = a;

//some code

cur_value = b;
```

Литералы - это явно заданные значения в коде программы — константы определенного типа.

0b(2), 0(8), 0x(16)

Экранированные символы используются для задания символов в тех позициях, где реальный символ нарушает синтаксис программы.

Символ `\b` позволяет нам удалить последний символ в строке вывода

Символ возврата каретки `\r` позволяет нам вернуть курсор к началу строки вывода и отображать новую информацию так, как будто ранее в этой строке ничего не было.

Символ табуляции в строке обозначается escape-последовательностью `\t` и является аналогом четырех пробелов.

`\n` - перенос строки

И для обозначения, что текст необходимо начать печатать с новой *страницы* использовался символ `\f`

Работа с переменными. Декларация. Инициализация. Присваивание.

Имя переменной должно состоять из символов Unicode. Оно не должно совпадать с любым из ключевых слов языка Java, а также с логическими константами true и false. Две переменных не могут иметь одинаковые имена, если они находятся в одной области видимости.

Переменная — это именованный указатель на фрагмент памяти, содержащий значение данных.

Декларация и инициализация - тип_данных имя_переменной [= значение_по_умолчанию] ;

Также, значение переменной можно изменять в любой момент, используя **оператор присваивания**. Например:

```
int x = 10;

x = 20;
```

Инструкции ветвления (if-else, switch) и циклов (do, while, for).

Цикл с постусловием

```
do {
  op;
```

```
} while (boolean expr);
```
Цикл с предусловием
```
while (boolean expr) {
  op;
}
```
Цикл с параметром
```
for (init; expr; incr) {
  op;
}
```

Цикл по элементам
```
Цикл iterable — массив или коллекция с элементами  типа T
*/
for (T x : iterable) {
  op;
}
```
Ветвление
```
if(boolean expr) {
  op1;
} else {
  op2;
}
```
Множественный выбор
```
switch(int/string expr){
  case const1:
   op1;
  case const2:
   op2;
   break;
  default:
   op3;
}
```
Чем отличаются массивы от коллекций?
У коллекции есть такие свойства:
-Добавление элемента в список
-Вставка элемента в середину
-Поиск элемента в строке
-Удаление элемента из списка

Операторы и выражения в Java. Особенности вычисления, приоритеты операций.

Постфиксные операторы (x++, x--) сначала возвращают исходное значение, после чего увеличивают или уменьшают его

Унарные операции

- унарный минус "-" – меняет знак числа или выражения на противоположный;

- унарный плюс "+" – не выполняет никаких действий
- побитовое дополнение "~" (только для целых) – инвертирует все биты поля числа (меняет 0 на 1 и 1 на 0);
- префиксный инкремент "++" и декремент "--" - сначала вычисляет инкрмент/дикремент, затем возвращает его значение

Приоритеты операторов

постфиксные операторы	`expr++ expr-- . ::`
унарные операторы	`++expr --expr +expr -expr ~ !`
операции с типами	`new (cast)expr`
умножение/деление	`* / %`
сложение/вычитание	`+ -`
операторы сдвига	`<< >> >>>`
операторы отношения	`< > <= >= instanceof`
операторы равенства	`== !=`
поразрядное И	`&`
поразрядное искл. ИЛИ	`^`
поразрядное ИЛИ	`|`
логическое И	`&&`
логическое ИЛИ	`||`
условный оператор	`? :`
операторы присваивания	`= += -= *= /= %= >>= <<= >>>= &= ^= |=`
стрелка лямбда	`->`

`>>`	`a >> b`	**сдвиг вправо**	Сдвигает биты числа a, на b разрядов вправо.
`<<`	`c << d`	**сдвиг влево**	Сдвигает биты числа c, на d разрядов влево.
`>>>`	`a >>> 2`	**сдвиг вправо с заполнением нулем**	Сдвигает биты числа a, на 2 разряда вправо.

Математические функции в составе стандартной библиотеки Java. Класс java.lang.Math.

Класс Math располагается в пакете java.lang и предоставляет набор статических методов для осуществления ряда различных математических вычислений.

Метод Math.random() возвращает случайное число с плавающей точкой (интервал [0, 1)). Также используется приведение типов, например, (int) (Math.random()*((max-min)+min)

Если использовать библиотеку java.util.Random, то используются методы nextInt(), nextDouble() и т.д

Подпрограммы, методы, параметры и возвращаемые значения.

Подпрограмма - это отдельная часть программы, имеющая имя и решающая свою отдельную задачу. Она может быть запущена (вызвана) из основной программы по указанию имени.

Подпрограммы бывают двух типов - **процедуры и функции**.

Подпрограммы-процедуры **выполняют некоторые действия, например, выводят результат на экран в определенном виде**. Простой пример, оператор printf()

Подпрограммы-функции **возвращают результат** (число, символьную строчку и т.д.), который мы можем использовать в основном программе.

Метод — это подпрограмма, которая может принимать необходимые параметры и возвращать результат своей работы.

Параметр - это переменная, от значения которой зависит работа подпрограммы. Имена параметров перечисляются через запятую в заголовке подпрограммы. Перед параметром записывается его тип.

Аргумент - это значение параметра, которое передается подпрограмме при ее вызове.

Форматированный вывод числовых данных.

Методы для форматирования строки

1. String.format()
2. System.out.format()
3. System.out.printf()

%[ширина][.точность] спецификатор типа

Например, System.out.format(“%7.5f”, 3.19342521)

Ширина - положительное число, если перед числом стоит 0, то недостающие символы будут заполнены нулями. Иначе они будут заполнены пробелами.

Точность - кол-во знаков после запятой