Хоренко Евгений 10/22 = 3

1) Что из перечисленного является правильным идентификатором?

a) false
b) _object
c) a-class
d) 2phase

2) Какова ширина базового типа int в Java:

a) 2 байта
b) 4 байта
c) 8 байт
d) зависит от архитектуры процессора

3) В каком из примеров 2-мерный массив объявлен НЕПРАВИЛЬНО:

a) int m[ ][ ] = new int [10][5];
b) int m[10][5];
c) int m[ ][ ] = {{1,2},{3,4}};
d) int m[ ][ ] = new int [10][ ];

4) Какое из указанных ниже преобразований типа в зыке Java может быть выполнено автоматически

a) byte -> int
b) char -> byte
c) char -> short
d) short -> char

5) Выберите верное утверждение:
Если член класса объявлен с модификатором protected, доступ к нему возможен:

a) только из непосредственных подклассов
b) только из всех классов, находящихся в том же пакете
c) только из подклассов, находящихся в том же пакете
d) из классов, находящихся в том же пакете, и непосредственных подклассов, находящихся в других пакетах

6) Как можно уничтожить объект в Java:

a) вызвать Runtime.getRuntime().gc();
b) вызвать метод finalize() у объекта
c) этого нельзя сделать вручную
d) вызвать деструктор у объекта

7) Класс Dog создан в пакете animals, а класс Peter - в пакете humans. Как необходимо написать конструкцию import для того, чтобы класс Peter мог использовать объекты Dog?

a) import animals.Dog;
b) import Dog.animals;
c) import *.animals;
d) import animals.Dog.*;

8) Переопределением метода называется:

a) создание в подклассе метода, совпадающего по имени с методом суперкласса
b) создание в одном классе двух методов, совпадающих по сигнатуре
c) создание в подклассе метода, совпадающего по сигнатуре с методом суперкласса
d) создание в подклассе метода, совпадающего по имени, но не совпадающего по сигнатуре с методом суперкласса

9) Выберите верное утверждение:
Класс, унаследованный от абстрактного:

a) не может быть абстрактным
b) должен реализовывать все методы абстрактного класса, но не может иметь собственных методов
c) должен реализовывать все методы абстрактного класса либо сам быть объявлен как абстрактный
d) должен реализовывать хотя бы один метод абстрактного класса, при этом может содержать и собственные методы

10) В каких случаях компилятор создает в Java-классе конструктор по умолчанию:

a) всегда
b) если в классе нет конструктора без параметров
c) если в классе нет ни одного конструктора
d) никогда

11) Сколько интерфейсов может быть реализовано в Java-классе:

a) 1
b) 2
c) зависит от типа класса
d) любое количество

12) Дан код:

```
public class EnclosingOne {
   public class InsideOne{}
}
public class inertest {
   public static void main (String[]args){
      EnclosingOne eo = new EnclosingOne();
      //any code here
   }
}
```

Какое выражение в выделенной строке создаст экземпляр вложенного класса?

a) InsideOne ei = eo.new InsideOne();
b) EnclosingOne.InsideOne ei = eo.new InsideOne();
c) InsideOne ei = EnclosingOne.new InsideOne();

13) Дан класс для хранения комплексных чисел:

```
public class Complex {
   private int Re, Im;
   public Complex() {    }
   public Complex(int Re, int Im) {
      this.Re = Re; this.Im = Im;
   }
}
```

Какой метод класса Object надо переопределить в классе Complex чтобы можно было выводить комплексные числа на экран методом System.out.println() ?

a) Object clone()
b) boolean equals(Object object)
c) String toString()
d) int hashCode()

14) Какие типы исключений НЕОБЯЗАТЕЛЬНЫ к обработке в языке Java:

a) унаследованные от класса IOException
b) унаследованные от класса Error либо от класса RuntimeException
c) сгенерированные программно
d) унаследованные от класса Exception но не от RuntimeException

15) Будет ли компилироваться и работать следующий код:

```
try
{ …}
catch (Exception obj)
{ … }
catch (IOException obj)
{ …}
```

a) не будет компилироваться т.к. класс Exception является абстрактным
b) будет компилироваться и работать
c) не будет компилироваться т.к. IOException является подклассом Exception и второй блок catch – недостижимый код
d) будет компилироваться, но второй блок catch никогда не сработает, т.к. IOException является подклассом Exception

16) Как организовать сортировку значений в коллекции?

a) Переопределить метод compare() в интерфейсе Collection
b) Создать перегруженную версию add(), которая будет выполнять сортировку при добавлении элемента
c) Реализовать интерфейс Comparator и передать экземпляр реализующего класса в конструктор коллекции
d) Любым из перечисленных способов

17) Будет ли компилироваться и работать код:

```
class Point <T>
{   T x, y;
    Point () { x=0; y=0; }
    int metod()  { T obj = new T();   }
}
```

a) Да, будет
b) Не будет компилироваться, т.к. у класса T нет конструктора
c) Не будет компилироваться, т.к. нет информации о параметризованном типе в runtime
d) Будет компилироваться, но выбросит исключение при выполнении

18) Пусть есть следующая иерархия классов:

```
          |---------->WildAnimals--------|--------->Wolf
          |                              |--------->Tiger
Animals ---|
          |---------->DomesticAnimals-----|--------->Cat
                                          |--------->Dog
```

и параметризованная коллекция
HashSet <WildAnimals>Zoo = new HashSet <WildAnimals>();
Какой из вызовов метода add() будет правильным:

a) boolean b = Zoo.add(new Wolf());
b) boolean b = Zoo.add(new Animals());
c) boolean b = Zoo.add(new Cat());
d) boolean b = Zoo.add(new DomesticAnimals());

19) Какой из интерфейсов предоставляет возможность хранить объекты в виде пары "ключ-значение"?

a) java.util.Map
b) java.util.List
c) java.util.Set
d) java.util.SortedSet

20) В каком случае ThreadPoolExecutor будет создавать новый поток при поступлении новой задачи?

a) всегда
b) если потоков меньше, чем corePoolSize либо очередь задач полна
c) если число потоков не превышает maximumPoolSize
d) на усмотрение планировщика JVM

21) Члены класса с какими модификаторами НЕ подлежат сериализации? (ДВА правильных ответа)

a) protected
b) static
c) final
d) transient

22) Что выведет на консоль следующий код:

```
public class Exam
{
   public static void main(String[] arg)
   {
      Object obj = null;
      String str = new String("str");
      str = (String) obj;
      obj = new String("obj");
      System.out.print(obj + ", " + str);
   }
}
```

a) null, null
b) null, str
c) obj, null
d) str, obj