Решение задачи в среде Eclipse

Условие задачи.

Кандидаты проходят испытание, включающее в себя два теста, которые оцениваются целым числом в диапазоне от 0 до 100 включительно. Считается, что испытание пройдено, если суммарный результат не менее заранее заданного числа.

Создать массив объектов и вывести его содержимое на консоль.

Формализованное условие.

Создать пакет by.gsu.asoilab и определить в нем класс, описывающий испытание кандидата.

Поля класса:

- аккаунт кандидата,
- результат теста 1,
- результат теста 2,
- проходной балл (константа класса).

Конструкторы:

- конструктор по умолчанию,
- конструктор общего вида.

Методы:

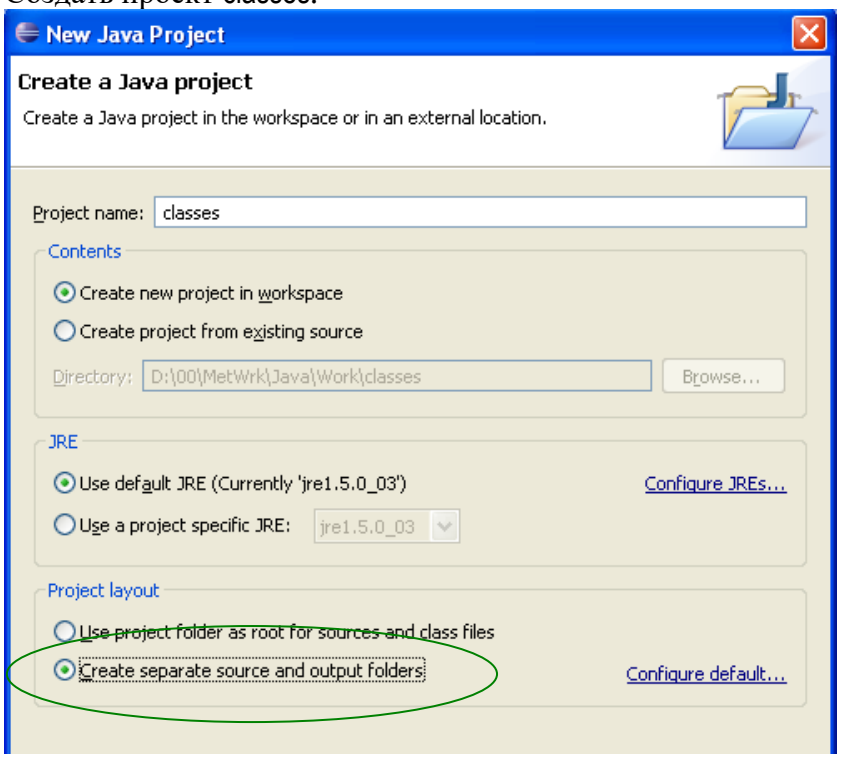
- геттеры/сеттеры,
- toString( ) –преобразование объекта в строку в формате csv: поля класса и итог испытания, разделенные символом ";".
- result( ) – сумма набранных баллов,
- isPassed( ) – итог испытания.

Создать раннер в пакете по умолчанию, где:

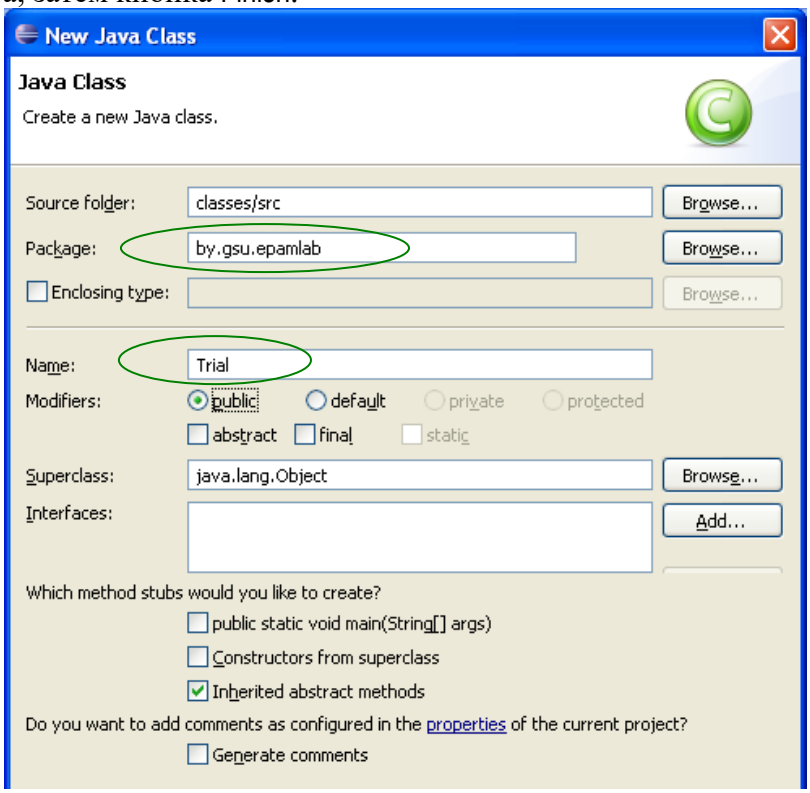
1. Создать массив из объектов.
2. Вывести содержимое массива на консоль.

Этапы решения задачи.

Создать проект classes.



Создать класс Trial в пакете by.gsu.epamlab. Щелчок правой кнопкой мыши по имени проекта и выбрать пункты меню New – Class. В диалоговом окне ввести имя пакета и имя класса, затем кнопка Finish.



Результат

```
package by.gsu.epamlab;

public class Trial {

}
```

Определить поля.

```
public class Trial {
	private String name;
	private int mark1;
	private int mark2;
	private static final int PASS_MARK = 120;
}
```

Создание конструкторов.

Выполнить пункт меню Source – Generate Constructors from Superclass и в диалоговом окне OK.

Выполнить пункт меню Source – Generate Constructor using Fields и в диалоговом окне OK.

Результат

```
public class Trial {
	private String name;
	private int mark1;
	private int mark2;
	private static final int PASS_MARK = 120;
	public Trial() {
		super();
		// TODO Auto-generated constructor stub
	}
	public Trial(String name, int mark1, int mark2) {
		super();
		this.name = name;
```

```
            this.mark1 = mark1;
            this.mark2 = mark2;
        }
}
```

Создание геттеров/сеттеров.

Выполнить пункт меню Source – Generate Getters and Setters и в диалоговом окне установить чекбоксы напротив полей name, mark1, mark2, а затем OK.

Результат – должны добавиться методы

```
    public int getMark1() {
            return mark1;
    }
    public void setMark1(int mark1) {
            this.mark1 = mark1;
    }
    public int getMark2() {
            return mark2;
    }
    public void setMark2(int mark2) {
            this.mark2 = mark2;
    }
    public String getName() {
            return name;
    }
    public void setName(String name) {
            this.name = name;
    }
```

Далее класс создается в полуавтоматическом режиме. Достаточно набрать символы ключевого слова, поля, метода, переменной и т.п., нажать Ctrl + пробел, появляется список из которого можно делать правильный выбор.

Создать вспомогательный метод, вычисляющий сумму набранных баллов. Метод полезный, но он будет использоваться только внутри класса. Поэтому его надо объявить приватным.

```
    private int result() {
            return mark1 + mark2;
    }
```

Создать метод, определяющий итог испытания. Метод будет вызываться из других классов. Поэтому его надо объявить публичным. При этом он использует код предыдущего приватного метода.

```
    public boolean isPassed() {
            return result() >= PASS_MARK;
    }
```

Создание метода toString().

Выполнить пункт меню Source – Override/Implement Methods и в диалоговом окне установить чекбокс напротив поля toString(), затем OK.

Результат

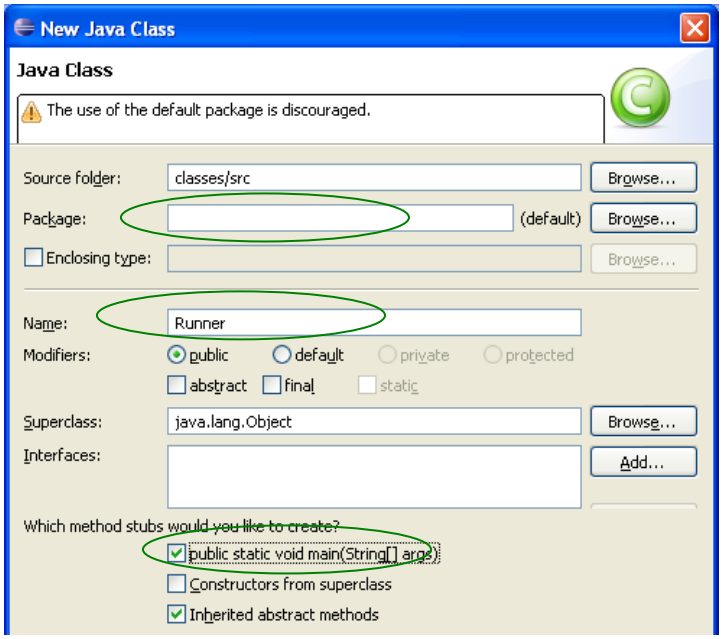
```
    @Override
    public String toString() {
            // TODO Auto-generated method stub
            return super.toString();
    }
```

Изменить тело метода.

```
    @Override
    public String toString() {
            return name + ";" + mark1 + ";" + mark2 + ";" + isPassed();
    }
```

Информационный класс создан.

Создание раннера в пакете по умолчанию, т.е. поле пакета в диалоговом окне оставить пустым, и установить чекбокс напротив метода main( ).



Результат

```
public class Runner {
	/**
	 * @param args
	 */
	public static void main(String[] args) {
		// TODO Auto-generated method stub
	}
}
```

Внутри метода main( ) ввести Trial, установить курсор внутри имени класса и нажать Ctrl + Shift + M. В начале файла добавится оператор импорта.

```
import by.gsu.epamlab.Trial;
```

Далее определить и инициализировать массив.

```
		Trial[] trials = {
				new Trial("Sakovich", 45, 93),
				null,
				new Trial("Zhylinsky", 51, 35),
				new Trial()
			};
```

Написать цикл вывода элементов массива.

```
		for (Trial trial : trials) {
			System.out.println(trial);
		}
```

При наличии ошибок отладить программу и получить результат.

```
Sakovich;45;93;true
null
Zhylinsky;51;35;false
null;0;0;false
```

Протестировать на различных исходных данных (элементах массива).