# Многопоточность в Java 1

# Задача1 (10)

Ответьте на следующие вопросы:

1. Что такое поток (Thread) и чем отличается от процесса (Process)?
2. Что такое упреждающее планирование (preemptive scheduling), разделение времени (time slicing), Совместное планирование (cooperative scheduling)?
3. Приведите примеры, где многопоточность используется или может быть использована?
4. Каковы различные этапы жизненного цикла потока в Java? Какие методы можно использовать для изменения состояния потока?
5. Что такое демон-поток в Java?
6. Что такое пул потоков (Thread Pool) в Java?
7. Что такое группа потоков (Thread Group)?
8. Что такое реентерабельный метод (reentrant method)?
9. Для чего используется ключевое слово synchronized?
10. Для чего используются методы wait(), notify() и notifyAll() в классе Object?

Вы должны ответить на все вопросы в отдельном pdf-файле (5 баллов). Во время защиты вам будет задано 5 вопросов из них (5 баллов). Без pdf-файла со всеми ответами домашнюю работу никак не защитить.

# Задача2 (5)

Завершите код, чтобы получить следующий результат:

Text

Description automatically generated

**Файл Main.java**

|  |
| --- |
| package org.example;  public class Main {  public static void main(String[] args) {  System.*out*.println("The different states of a thread are: ");    /\* Add code here \*/  for(Thread.State state:states)  {  System.*out*.print(state+" ");  }  System.*out*.print("\nLet's get practical:\n");   MyThreadRunnable t=new MyThreadRunnable("Thread States");  t.start();  //wait 500 milliseconds for the thread to die  /\*Add code here\*/   t.printState();  t.join(3000);  t.printState();  //Notify thread to wake up  /\* Add code here \*/   t.printState();  //wait forever for the thread to die  /\* Add code here \*/   t.printState();   } } |

**Файл MyThreadRunnable.java**

|  |
| --- |
| package org.example;  public class MyThreadRunnable implements Runnable{  private Thread t;  private final String name;  private final Object obj;  public MyThreadRunnable(String name)  {  this.name=name;  obj=new Object();  }  @Override  public void run() {  printState();  try {  //Make thread sleep for 2000 milliseconds  /\*Add some code here\*/   } catch (InterruptedException e) {  throw new RuntimeException(e);  }  synchronized (obj) {  try {  //make the thread wait until it is notified  /\*Add some code here\*/   } catch (InterruptedException ex) {  throw new RuntimeException(ex);  }  }  printState();  }  public void start()  {  t=new Thread(this,name);  printState();  t.start();  }   public void printState()  {  //print the current state of the thread  /\*Add some code here\*/  }   public void join() {  try {  t.join();  } catch (InterruptedException e) {  throw new RuntimeException(e);  }  }  public void join(int millis) {  try {  t.join(millis);  } catch (InterruptedException e) {  throw new RuntimeException(e);  }  }   public void threadNotify()  {  synchronized (obj) {  obj.notify();  }  } } |

# Задача3 (5)

Напишите программу с точно такими же выходными данными и классами. Однако MyThreadRunnable должен наследовать класс Thread. Почему иногда лучше реализовать интерфейс Runnable вместо наследования класса Thread?