# Описание работы библиотеки расчета эффективности использования времени сотрудника

Время работы сотрудника по табелю включает нахождение сотрудника на рабочем месте, поэтому важно понимать, сколько времени он тратит на рабочие задачи, где есть простой из-за других сотрудников или недостатков планирования, а где просто нехватка задач для максимально эффективной загрузки.

Если у сотрудника нет задач в заданный промежуток табеля учета рабочего времени, то такое время обозначается как «Время простоя без задач».

Если у сотрудника появляется сдвиг задачи при смещении сопутствующих задач других сотрудников, то такое время называется «Время простоя по вине другого сотрудника»

Если у сотрудника образовался простой, но при этом имеются запланированные задачи, к выполнению которых он не может приступить, то такое время называется «Время простоя из-за недостатков планирования».

*Время простоя = Время простоя без задач + Время простоя по вине другого сотрудника + Время простоя из-за недостатков планирования.*

*Коэффициент эффективности работы сотрудника = 1 - ( Время простоя / Общее время работы по табелю).*

Вход:

* табель учета рабочего времени сотрудников (в качестве примера данных можно использовать Timesheets.xlsx),
* запланированные задачи за учетный период.

Выход:

Метод расчета эффективности использования времени сотрудника должен возвращать массив со списком следующих показателей:

* Коэффициент эффективности работы сотрудника (в процентах, с округлением до сотых),
* Время простоя в секундах,
* Время простоя без задач в секундах,
* Время простоя по вине другого сотрудника в секундах,
* Время простоя из-за недостатков планирования в секундах.

## Требования к именованиям и форматам:

|  | C# | Java | Python |
| --- | --- | --- | --- |
| Библиотека классов | WorkingTimeEfficiency{NN}Lib.dll | WorkingTimeEfficiency{NN}Lib.jar | WorkingTime Efficiency{NN}Lib |
| Название класса | WorkingTime | WorkingTime | WorkingTime |
| Название метода | CalculateEfficiency() | CalculateEfficiency() | CalculateEfficiency() |
| Статичность метода | да | да | да |
| Входящие обязательные параметры | List<WorkingDay> workingDays  List<Task> tasks | List<WorkingDay> workingDays,  List<Task> tasks | workingDays: List[WorkingDay],  tasks: List[Task], |
| Возвращаемые параметры | double[] | double[] | List[float] |
| Класс WorkingDay | public class WorkingDay  {  public DateOnly Date { get; set; }  private int hours;  public int Hours  {  get => (Status == StatusDay.**Working** || Status == StatusDay.**PreWorking**) ? hours:0;  set => hours = (Status == StatusDay.**Working** || Status == StatusDay.**PreWorking**) ? value : 0;  }  public StatusDay Status { get; set; }  }  public enum StatusDay  {  **Working**, *// Рабочий день*  **PreWorking**, *// Предпраздничный день*  SickLeave, *// Больничный*  DisrespectfulSash, *// Отсутвие по неуважительной причине*  Holiday *// Отпуск или выходной нерабочий день*  } | @Data  @Builder  public class WorkingDay {  private final LocalDateTime date;  private final int hours;  private final StatusDay status;  }  public enum StatusDay {  WORKING(0), *// Рабочий день*  PREWORKING(1), *// Предпраздничный день*  SICKLEAVE(2), *// Больничный*  DISRESPECTFULSASH(3), *// Отсутвие по неуважительной причине*  HOLIDAY(4); *// Отпуск или выходной нерабочий день ;*  private final int status;  StatusDay(int status) {  this.status = status;  }  public int getStatusDayAsInt() {  return status;  }  public static StatusDay convertIntToStatusDay(int status) {  for (StatusDay st : StatusDay.values()) {  if (st.getStatusDayAsInt() == status) {  return st;  }  }  return null;  }  } | class WorkingDay:  date: datetime  hours: int  status\_day: StatusDay  class StatusDay(enum.IntEnum):  Working = 0 *# Рабочий день*  PreWorking = 1 *# Предпраздничный день*  SickLeave = 2 *# Больничный*  DisrespectfulSash = 3 *# Отсутвие по неуважительной причине*  Holiday = 4 *# Отпуск или выходной нерабочий день* |
| Описание класса Task | **public class Task**  **{**  **public string Guid { get; set; }**  **public string FullTitle { get; set; }**  **public string Description { get; set; }**  **public DateTime BeginDateTime { get; set; }**  **public DateTime EndDateTime { get; set; }**  **}** | **@Data**  **@Builder**  **public class Task {**  **private final String guid;**  **private final String FullTitle;**  **private final String Description;**  **private final LocalDateTime beginDateTime;**  **private final LocalDateTime endDateTime;**  **}** | class Task:  guid: str  full\_title: str  description: str  begin\_datetime: datetime  end\_datetime: datetime |

## 

## Пример:

Для должности Инженер-программист табель учета рабочего времени за июнь выглядит так:

| Должность | Май | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Инженер-  программист | 8 | 8 | 8 | 8 | 8 |  |  | 8 | 8 | 8 | 8 | 8 |  |  | 8 | 8 | 8 | 8 | 8 |  |  | 8 | 8 | 8 | 8 | 8 |  |  | 8 | 8 | 8 |

Суммарное рабочее время по табелю = 184 ч.

За учетный период Инженер-программист выполнил следующие задачи:

| Наименование задачи | Статус | Дата создания | Дата изменения | Крайний срок | Предшествующая задача | Дата начала работы | Дата окончания работы |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Создание интерфейсов и наборов прав пользователей | На проверке | 2022-10-09 00:00:00 | 2023-01-31 00:00:00 | 2023-02-01 00:00:00 | ВОДОКАНАЛ-06 | 2023-05-01 00:00:00 | 2023-05-20 00:00:00 |

За заданный период у Инженера программиста была только одна задача, поэтому потери по времени - Время простоя без задач.

*Время простоя без задач = 8\*8 = 64 ч.*

*Время простоя = Время простоя без задач + Время простоя по вине другого сотрудника + Время простоя из-за недостатков планирования = 64 ч.*

*Коэффициент эффективности работы сотрудника = 1 - (64/184) = 0,65217391 ~ 65,22%*

Метод должен вернуть массив следующих значений: **[65.22 ,230400, 230400, 0, 0]**