排课算法入参（schedulingQeryDto）

```
private List<AlgorithmsQueryDto> list;
```

参数：AlgorithmsQueryDto

```
int amCourseCount;//上午课时
int pmCourseCount;//下午课时
int daysPerWeek = 5;//每周上课天数
int scheduleType;/* 0：普通排课 or 定二走一，1：大走班排课*/
List<TimeDto> timeList; //时间列表
List<ClassDto> classList;//班级列表
List<RoomDto> roomList ; //教室列表
List<CourseDto> courseList; //课程列表
List<TeachingPlanDto> teachingPlanList;//任课计划列表
List<CourseTeacherDto> courseTeacherList;//教师列表
List<TeachingPlanDto> preTeachingPlanList;//预排课列表
List<RuleDto> rulePriorityList = new ArrayList<>();//优先级
Map<CourseDto,CourseDto> mutualExclusionMap = new HashMap<>();//互斥课程
Map<CourseDto,CourseDto> singleDoubleMap = new HashMap<>();//单双周
long algorithmStartTime; //算法启动时间
SelectQueryDto selectQueryDto = new SelectQueryDto();//业务使用
Map<Integer,List<Integer>> mutualGroupMap;//合并课程组，具体闭周
```

参数：RuleDto(已废弃、规则优先级)

```
int ruleId
int rulePriority（规则优先级）
boolean isValid（生效）
```

参数：CourseTeacherDto

```
CourseDto course（学科信息）
List<TeacherDto> list;（教师列表）
```

参数：TeachingPlanDto

```
int typeSecondary ; //业务数据，算法没用到？？？
int[] planType = new int[Constant.PLAN_TYPE_MAX]; //标记课程类型{普通、预排课、分层走班、合班排课、单双周}
int currentCourseScheduleTimes; //当前课程是本班级的第几节课程
int scheduleLevel; /*处理排课级别，用于控制排课优先级   * 当前逻辑：   * 1、必须排课   * 2、分层走班、合班、单双周   * 3、其他课程   * */ ClassDto classDto;
TeacherDto teacherDto
CourseDto courseDto
RoomDto roomDto
TeacherDto teacherDto2//双周教师信息
CourseDto courseDto2//双周课程信息，如果为空表示两周课程一致
RoomDto roomDto2//双周教室信息
TimeDto timeDto = new TimeDto();
```

参数：TimeDto

```
int day;
int section;
int[] type  = new int[Constant.CLASS_TYPE_MAX];//标识班级类型，单科班、两科班、三科班、四科班，1+1班
int[] teachingType = new int[Constant.GROUP_MAX];//标识不同课程组上课可用时间{预留、课程组1、2、3、4、5、6、7、8}
int index;//用于交叉流程，记录交叉前的timeIndex
double selectThread= Constant.SELECT_THREAD;//选择阈值
Map<Integer,Integer> selectFlagMap = new HashMap<>();//
暂时没用
```

参数：ClassDto

```
int classId;
String className;
int studentCount;
List<StudentDto> students;
int classType;/*1:行政，2：教学*/

/*   * 教学班-> 0：选考，1：学考
* 行政班-> 0  大走班，  2  定二走一，  3  优先三科成班，  4  定四走二    * */
int teachingClassType;
int groupId;    //行政班：合班和分层走班，业务传入;
                教学班：标识课程组groupId
int subjectCode;   //大走班分班，用于基因编码
int sortId;  //班级冲突排序（已废弃）
int level;//班级分层，主要用于教师按层分配
```

参数：RoomDto

```
int roomId;
String roomName;
int roomType;    //教室类型，用于功能教室
int releaseCourseCount;    //剩余课时数
int maxCourseCount;   //最大课时数
Map<Integer,Integer> classesCode;   /*教室分层情况，1表示已排课程，0表示未排课程*/
Map<Integer,ClassDto> layerGroup /*记录分组使用教室情况*/
Map<Integer,ClassDto> courseGroup/*课程组<teachingClassType,classId(教学班)>*/
int singleDouble;   /*单双周标记*/
int source;   / *0表示业务创建，1表示算法创建，暂时仅记录，不使用*/
Map<TimeDto,Integer> courseInfo;/*教室上课详情：教室课表*/
```

参数：CourseDto

```
int courseId;
String courseName;
int courseCount;  /*周课时*/ int roomType;    //教室类型，用于功能教室，和RoomDto对应
int groupType=-1; /*排课规则*/ int maxCountPerDay=-1; //每天最大课时规则
int maxCountPerSection=-1; //同时最大授课数规则
int noTeacher = -1; //不分配教师规则
List<RoomDto> roomList = new ArrayList<>();//功能教室列表
List<TimeDto> rejectTimeList = new ArrayList<>(); //课程拒绝排课时间列表
List<TimeDto> mustTimeList = new ArrayList<>();//课程必须排课时间列表
```

参数：TeacherDto

```
int personId;
String personName;
int courseId;/*胜任课程*/
int isVirtual = 0; //虚拟教师 -1 int maxCourseCount; /*最大周课时*/
int maxSelectCourseCount; /*最大选考课时*/
int releaseCourseCount;   /*剩余周课时*/
int mutexesCount;    /*互斥课程数*///
int singleDouble;  /*单双周标记*/
int courseCount; /*教师周代课数*/
int preCourseCount;     /*教师预排课数量*/
int groupId;   /*教师分层*/
Map<Integer,ClassDto> courseGroup/*课程组<teachingClassType,classId(教学班)>*/
Map<Integer,ClassDto> layerGroup/*教师分层代课情况，1表示已带课程，0表示未带课程*/
private Map<TimeDto,Integer> courseInfo = new HashMap<>(); /*教师上课详情*/
```

分班算法入参（MobileLearningQuery）

```
private int minStudentCount;
private int maxStudentCount;
private int classRoomCount;
private List<TimeDto> times;
private List<RoomDto> rooms;
private List<MobileLearningTeacherDto> teachers;
private List<MobileLearningSubject> mobileLearningSubjects;
private List<MobileLearningSubject> mobileLearningMinorSubjects;
private Map<String,VolunteerResult> volunteerResultMap;
private Map<String,Integer> teacherAvgCourseMap;
private Map<String,Map<String,VolunteerResult>> volunteerLevelResult;
private SelectQueryDto selectQueryDto;
private MobileLeaningCallBackIntercace mobileLeaningCallBackIntercace;
```