**procedure** TIMESLOTROOMALLOCATION (*Students, Exams, Timeslots, Departments, Rooms*)

S: sizeof (*Students*), E: sizeof (*Exams*), T: sizeof (*Timeslots*), D: sizeof (*Departments*), R: sizeof (*Rooms*)

*timetable* new int [N][S]

*sortedRooms* new PriorityQueue

*availableRooms* Rooms

*availableTimeslots* Timeslots

**Step 1: Grouping Students into Departments for every Exam**

For Each (exam ei in *Exams*) {

For Each (department dj in *Departments* {

For Each (student sk enrolled in ei) {

If (*department* [sk] = dj) dj.*examEnrollment*[ei].add(sk)

}

}

}

**Step 2: Timeslot and Room Allocation**

For Each (exam ei in *Exams*) {

**Step 2.1: Allocate Random Timeslot to Exam**

If (sizeof (*availableTimeslots*) > 0) {

slot random(T)

E.setTimeslot(slot)

availableTimeslots.remove(T)

}

**Step 2.2: Allocate Student to Closest Room**

For Each (department dj in *Departments* {

For Each (Room rj in *Rooms*) rj.setDistanceToDepartment(dj.location)

*sortedRooms* *Rooms*

For Each (student sk from dj enrolled in ei){

*currentRoom* *sortedRooms*.peek()

If (*currentRoom*.freeSeats > 0){

sk.room *currentRoom*

*timetable*[ei][sk] *currentRoom*

}else *sortedRooms*.remove(*currentRoom*)

}

}

}

return *timetable*

**end procedure**

**procedure** TIMESLOTROOMALLOCATION (*Students, Exams, Timeslots, Departments, Rooms*)

S: sizeof (*Students*), E: sizeof (*Exams*), T: sizeof (*Timeslots*), D: sizeof (*Departments*), R: sizeof (*Rooms*)

*timetable* new int [N][S]

*sortedRooms* new PriorityQueue

*availableRooms* Rooms

*availableTimeslots* Timeslots

**Step 1: Grouping Students into Departments for every Exam**

For Each (exam ei in *Exams*) {

For Each (department dj in *Departments* {

For Each (student sk enrolled in ei) {

If (*department* [sk] = dj) dj.*examEnrollment*[ei].add(sk)

}

}

}

**Step 2: Timeslot and Room Allocation**

For Each (exam ei in *Exams*) {

**Step 2.1: Allocate Random Timeslot to Exam**

If (sizeof (*availableTimeslots*) > 0) {

slot random(T)

E.setTimeslot(slot)

availableTimeslots.remove(T)

}

**Step 2.2: Allocate Student to Closest Room**

For Each (department dj in *Departments* {

For Each (Room rj in *Rooms*) rj.setDistanceToDepartment(dj.location)

*sortedRooms* *Rooms*

For Each (student sk from dj enrolled in ei){

*currentRoom* *sortedRooms*.peek()

If (*currentRoom*.freeSeats > 0){

sk.room *currentRoom*

*timetable*[ei][sk] *currentRoom*

}else *sortedRooms*.remove(*currentRoom*)

}

}

}

return *timetable*

**end procedure**