The problem of creating a valid timetable involves scheduling classes, teachers and rooms in such a way that no teacher, class or room is used more than once per period. For example, if a class must meet twice a week, then it must be placed in two different periods to avoid a clash. The timetable is to be distributed across a fixed number of periods per week. A class consists of a number of students. We will assume that students have already been grouped into classes.

Initially we consider classes to be disjoint, that is, they have no students in common. In this scheme, a correct timetable is one in which a class can be scheduled concurrently with any other class. Later in the paper this restriction will be relaxed to cater for high schools and Universities in which students can take many options. In each period a class is taught a subject. It is possible for a subject to appear more than once in a period. A particular combination of a teacher, a subject, a room and a class is called an element. An element may be required more than once per week. The combination of an element and a frequency is called a requirement. Thus, the timetabling problem can be phrased as scheduling a number of requirements such that a requirement, teacher, class or room does not appear more than once per period.

There can be a timetable consisting of 6 periods (at 9 am, at 10 am, at 11 am, at 12 pm, at 2 pm, at 3 pm) per day, 5 days (monday, tuesday, wednesday, thursday, friday) per week, which means there can be a total of 30 periods per week. There will be a class clash or conflict if a timetable shows both (i) Class 1, Room 5, Teacher 1 (ii) Class 1, Room 9, Teacher 2 appearing in at 9 am Monday slot. But there is no class clash if (i) Class 1, Room 5, Teacher 1 appears in at 9 am Monday slot and (ii) Class 1, Room 9, Teacher 2 appears in at 10 am Monday slot because they are two different time slots. Similarly, there is a teacher conflict, if one teacher appears more than once in a time slot, and a room conflict, if one room appears more than once in a time slot.

It is possible to define an objective or cost function for evaluating a given timetable. This function is an arbitrary measure of the quality of the solution. A convenient cost function calculates the number of clashes in any given timetable. An acceptable timetable has a cost of 0.
The optimization problem becomes one of minimizing the value of this cost function. The cost of any period can be expressed as the sum of three components corresponding to a class cost, a teacher cost and a room cost. By using a sum, it becomes easy to weight the various costs so that one may be made more important that the others. In this way the optimization process can be guided towards a solution in which some types of clash are more important than others.

The class cost is the number of times each of the classes in the period appears in that period, less one if it is greater than zero. Thus, if a class appears no times or once in a period then the cost of that class is zero. If it appears many times the the class cost for that class is the number of times less one. The class cost for a period is the sum of all class costs. The same computation applies for teachers and rooms. The cost of the total timetable is the sum of the period costs.

In this assignment, you would run experiments varying the number the number periods per day. There might be 35 periods per week or 40 periods week. In addition, you would run experiments varying the weighting factors for class costs, teacher costs, and room costs. Initially, you can have equal weighting factors [1 1 1]. Later on, you can run experiments with [10 1 1], [1 10 1], [1 1 10].

Lab sections A1 and B1 are expected to run experiments using hill-climbing and its variants.
Lab sections A2 and B2 are expected to run experiments using local beam search and its variants.

DESCRIPTION OF DATA FILES:

The data files describe five timetabling problems: hdtt4, hdtt5, hdtt6, hdtt7, and hdtt8. The optimal objective function for each of these problems is zero clashes.

Each problem consists of three text files. For hdtt4 these files are: hdtt4list (contains the list of requirements expressed as English statements of the form Class C1 meets teacher T3 in room R4); hdtt4note (contains the dimensions of the problem, in this case 4 classes, 4 teachers, 4 rooms, 30 periods, and 120 requirements); hdtt4req (contains a requirements matrix extracted from the list of requirements).

Actually, hdtt4req is created from using the information mentioned in hdtt4list. Someone wishing to use these data sets need only consider the dimensions of the problem (given in hdtt4note), and the requirements matrix (given in hdtt4req), which is read as follows:

Suppose there are C classes, T teachers, V venues, and P periods. Then the first V rows of matrix indicate the number of times each class-teacher combination is to meet each other in venue 1 across the P periods. The next V rows indicate the number of times each class-teacher combination is to meet each other in venue 2 across the P periods, etc.

For example, in hdtt4req, the 3rd row contains a "6" in the last column. This means that class 3 must meet teacher 4 in room 1 six times. Similarly, row 5 column 2 contains a "5", meaning that class 1 must meet teacher 2 in room 2 five times.

The grouping of rows according to venu is shown below for hdtt4:

teacher 1 2 3 4

```
class 1, 2 2 1 2, room 1
class 2, 1 1 1 2, room 1
class 3, 1 1 1 6, room 1
class 4, 2 2 3 2, room 1
------------
class 1, 2 5 1 2, room 2
class 2, 0 4 3 2, room 2
class 3, 1 2 1 0, room 2
class 4, 2 2 1 2, room 2
------------
class 1, 2 1 1 2, room 3
class 2, 0 0 5 1, room 3
class 3, 2 1 4 1, room 3
class 4, 6 1 2 1, room 3
------------
class 1, 3 1 2 1, room 4
class 2, 1 4 1 4, room 4
class 3, 3 3 2 1, room 4
class 4, 2 0 1 1, room 4
------------
```

You can ignore the number of subjects mentioned in hdtt4note.txt. Also, you can ignore Importance of Blocks, Importance of preference, Importance of limits, Importance of relations, Importance of distribution mentioned in hdtt4list.txt.