CMU Fall 2010
Final Exam Schedule

Operations Research II Project

By Kweon Woo Jung, Min Jung Kim, Jisu Kwak, KangHa Lim

ABSTRACT

The paper investigates the optimal final exam schedule for Fall 2010 semester at Carnegie Mellon University. By optimal, we seek to minimize the number of students having a conflict or three consecutive exams within 24-hour period in their final exam schedule. This problem is a realistic problem and an interesting problem to approach at the same time. The fact that the final exam schedule is finalized almost a week or two before the actual final exam implies us the difficulty and complexity of this problem. We hope to explore this problem with sufficient but reasonable assumptions to enable us to apply the mathematical reasoning learnt in Operations Research class.

ASSUMPTIONS

In order to reduce the complexity of the problem, we made several assumptions that are reasonable so that we can approach the problem and apply the solution in reality at the same time. These assumption are categorized mainly in four categories: the number of students enrolled for fall semester in 2010 at Carnegie Mellon University, course schedules for the students, room information where the final exams are taken, and miscellaneous assumptions beside the first three assumptions.

Number of Students Enrolled

We first made an assumption on the number of students enrolled for fall semester in 2010 at Carnegie Mellon University for undergraduates. In order to do so, we looked up the undergraduate first-year admission statistics for year 2010-2011 so that we make our assumption on the number of students enrolled as realistic as possible. We rounded up the numbers of students enrolled in each department such as Carnegie Institute of Technology (CIT), College of Fine Arts (CFA), College of Humanities and Social Sciences (H&SS), Tepper School of Business (Tepper), H. John Heinz II College (IS), Mellon College of Science (MCS), and School of Computer Science (SCS). We assumed zero students for BHA/BSA/BCSA students since they comprise only about 0.01% of the total students. Hence, we came up with the total number of 6000 students, having 1500 students in each grade, Freshmen, Sophomore, Junior, and Senior, and distributed reasonable number of students following the statistics for 2010-2011 year (Appendix A).

Students' Course Schedule

Constructing course schedules for each student, we looked up the student catalogue on CMU website where suggested schedules and requirements are displayed for each major in each department (Appendix A). From there, we created schedules for each major in each department, considering only courses that are offered during the fall semester. In order to make our assumed students' schedules more realistic, we took a survey on which electives are most frequently taken by juniors and seniors for each major (Appendix A). We also assumed that students take four or five courses on average. Then we put our data together and created schedules for each major for each year from freshmen to seniors (Appendix A).

Room Information where Final Exams are taken

Next, we made an assumption on the capacity of the classrooms where the final exams will be taken. We researched on CMU website and obtained information on capacity of how many students can fit in each room at a maximum (Appendix A). We assumed that 35 classrooms including MM 103 are used for the final exams and the largest room came out to be UC McConomy and the smallest room came out to be Wean Hall 5304.

Miscellaneous

Beside three assumptions above, we assume that the final exams are taken for six days excluding the reading day, which is same as the current policy at CMU. Also, we assume that there are three exam periods per day. We assume that the schedule we receive is fixed before we make an optimal final exam schedule, which means our information on students' schedule for fall semester is the information after the add/drop day so that the course registration is fixed for the students. No two exams can be taken simultaneously in the same room although the room is not full; however, for over-capacity courses, multiple rooms are available for the final exams during the same time period. Lastly, students are asked to take the final exams in every other seat.

IP FORMULATION

As it is mentioned above, our goal of the project is to minimize the number of students who are taking three exams within a 24-hour period. When we minimize the number of students taking three consecutive exams within a 24-hour period, we need to take some constraints into account. The most important limitation we should consider is that a student is not able to take two or more exams on any exam period – no time conflict. Secondly, on any exam period, the number of exams on that period should be less or equal to the number of rooms available for exams. Thirdly, there should be no room conflicts. In other words, no two exams can be taken in the same room. Thirdly, on any exam period, the number of exams on that period should be less or equal to the number of rooms available for exams. Fourthly, the capacity of the room(s) for certain exam should be able to accommodate all students who take that exam. Before we write equations and constraints, the very first thing we need to do is to define variables that appear in the equations.

Defining Variables: Object Equation

$X_{i,g,k}$ is an indicator variable that specifies whether a student has three consecutive exams within 24-hour period. We note as one if a student's exam schedule applies to the above indicator variable condition (by this, we mean that a student has three exams in 24-hour period), and zero otherwise. The subscription *i* indicates major of a student. There are total 23 majors so *i* consists of 1 from 23 – $i \in [1, 23]$. *g* signifies grade level of a students; since typical college including Carnegie Mellon University divides students into freshmen, sophomore, junior and senior, g consists of 1 from 4 - $g \in [1, 4]$, and *k* denotes the time period in which an exam is taken. $S_{j,g,t}$ is the number of grade level *g*-students taking exam *j* at time *t*. Since there are 3 exam periods in one day – morning, afternoon, and evening- and all the exams are scheduled such that it finishes in 6 days, *t* consists of 1 from 18 - $t \in [1, 18]$ in which one indicates the first exam period of the first day, two indicates the second exam period of the first day and so on.

Before we construct our object equation, recall that we assumed that students in the same major take the same courses; hence, all the students of certain grade and of certain major have the same exam schedule. For example, let us assume that a freshmen student in mathematic major has programming exam on the first period of the first day, history exam on the second period and principles of economics exam on the third period. Then all the freshmen students in mathematics major have the same exam schedule as the student above. Now, if we multiply above two variables, $X_{i,g,k}$ and $S_{j,g,k}$, we obtain the number of students in major *i*, grade level *g* and exam *j* as a starting exam who has three exams in 24-hour period as a result. We must sum over all time periods, majors, and grade levels to acquire the total number of students who take three consecutive exams. By minimizing this sum, we achieve our aim of the object equation.

Defining Variables: Constraint Equations

We also used integer programming to represent constraints in mathematical formula. $m_{i,j,g}$ is an integer variable. It corresponds to one if a grade level *g* of major *i* take exam *j* at certain time *t* or zero otherwise. Since we defined such that no student is able to take two or more exams on any exam period, we should get zero or one when we sum $m_{i,j,g}$ over all *j* (there are total 126 exams so *j* consists of 1 to 126 – $j \in [1, 126]$). If we sum this variable over all *i,j* and *g*, and limit this summation to be less than 35 (the total number of rooms used for final exams), this inequality represents the constraint three – the number of exams held on any period should be less or equal to the number of rooms available for exams.

Not only students but rooms should have no conflicts – meaning that no two exams can be taken simultaneously in the same room. We created integer variable $r_{l,t}$ and defined to be one if room *l* is not used at time *t* or zero otherwise (the total of 35 rooms are used for final exams, so *l* consists of 1 to 34 – $l \in [1, 35]$).

Students are asked to take the final exams in every other seat. Let n_l represent the room capacity divided by two. Since we assumed that one or more rooms can be used for one exam, the number of seats of remaining rooms should be able to accommodate all students who take that exam. Hence $S_{j,g,t}$ summed over all g (the result gives all the students who take exam *j* and time *t*) should be less or equal the total number of seats of remaining rooms – $\sum_l n_l \cdot r_{l,t}$.

In *Defining Variables: Object Equation* section, we introduced variable $X_{i,g,k}$ but did not explicate in detail of the variable's composition. If a student a student has three exams in 24-hour period, we define the variable to be one. The kernel of the argument is the method of deciding whether a student has three exams in 24-hour period. If we sum $m_{i,j,g}$ over three exam periods, we are able to know the number of exams a student takes in 24-hour period.

Object and Constraint Equations in Mathematical Symbols

Minimize $\sum_{i}\sum_{g}\sum_{k}(X_{i,g,k} \cdot S_{j,g,k})$

Subject to

$$m_{i,j,g} = \begin{cases} 1 & \text{grade level g students majoring in i take exam j at certain } t - \text{period} \\ 0 & \text{else} \end{cases}$$

$i \in [1, 23], j \in [1, 126], g \in [1,4], t \in [1, 18]$

$$\sum_{j} m_{i,j,g} = 1 \text{ or } 0$$

$$\sum_{i}\sum_{j}\sum_{g} m_{i,j,g} \leq 35$$

$$r_{l,t} = \begin{cases} 1 & \text{room l is used at t} \\ 0 & \text{else} \end{cases}$$

$l \in [1, 35]$

$n_l = \text{room capacity}/2$

$S_{j,g,t} = \text{number of grade level g students taking exam j at time t}$

$j \in [1, 126]$

$$\sum_{g} S_{j,g,t} \leq \sum_{l}(n_l \cdot r_{l,t})$$

$$X_{i,g,k} = \begin{cases} 1 & \sum_{t=k}^{t=k+2} m_{i,j,g} = 3, k \in [1, 16] \\ 0 & \text{else} \end{cases}$$

We will use this Integer Programming Formula in Simple Optimal Algorithm later on.

INSIGHT FOR AN APPROACH

As Prof. Frieze has gently warned us, our problem is NP-hard, extraordinarily difficult problem to achieve the optimal solution. Hence it is reasonable and sufficient to pursue achieving a good solution rather than the optimal solution. Since we did not have a proficient knowledge of LINDO nor other ILP solvers, we have shifted our focus in generating an iterative optimality method using Java. Given an initial schedule, only bound to the room and reality constraint but ignoring the students' schedule constraint, we wish to generate an iterative optimality method that rebalances the schedule by decreasing the number of students having a schedule conflict.

In describing the Java approach in detail, it is assumed that the reader has a sufficient knowledge of Java. Of course, we are to provide non-Java explanation along with it.

Generating an Initial Schedule

In order for the iterative optimality schedule to work, we need an initial schedule to work with. An initial schedule is generated as follows:

First, store all the room, student and course data into the Java code using an ArrayList class in a sorted decreasing order. In other words, room, student and course information are stored in distinct memories inside Java so that we can access the data conveniently.

Starting from time 1(8:30 ~ 11:30 am of Exam Day 1), choose the course with largest student number. Then check whether this course can be assigned to the corresponding time, meaning iterate over all the empty class rooms and add up its capacity to compare it with the number of students taking a chosen course. If course can be assigned, fill in the rooms from largest to the smallest. If not, move on to the next time period for availability. Repeat such process with all the courses. Always start from time period 1.

Given the maximum room capacity throughout 18 periods and the sum of the number of students taking an exam, it is feasible to schedule classes, only bound to room constraints but ignoring the students' schedule conflict.

Repeating the above process with all the course lists, an initial schedule is generated only filling rooms up to time period 10 out of 18.

OPTIMALITY ALGORITHM

Identifying the Conflict

An initial schedule is set. Now we ought to find the optimality algorithm that reduces the number of students in conflict. To identify such a conflict, the following method is applied:

When assigning an exam in a room R at time T, the course variable stores the information within. In other words, once the exam is assigned to a specified room at specified time, the course now holds account of this information. Since no two different exams are scheduled in same room, due to the nature of constructing an initial algorithm, we only need to take an account of time conflict. Comparing the time periods of each student's exams, we can easily identify whether this student has a schedule conflict.

First, if two time periods are equal, it means two different exams a student is taking has been scheduled at same time period, which is a definite schedule conflict. We identify this conflict as a Concurrency conflict. For students having three or more exams, we need to identify whether three exams are scheduled within 24-hour period. We do such by discovering an interesting property such a combination provides. Iterate over each number, computing the absolute difference between the iterating number and the list of numbers. Increase the counter when we find the absolute difference to be less than three. If the resulting counter is greater than (five + size of the list of number) then the student has a 24-hour period conflict.

Iterating the initial schedule, we noticed total of 46 out of 80 student groups had a conflict in their course. Their schedule can be categorized in two distinct ones: Concurrency issue and 3-exams-in-24-hour issue. The prior being a student group has been assigned to take two or more exams at same time period, which is realistically impossible. The latter is that three exams are within the range of 24-hour period for student, which is very stressful and is ought to be avoided.

Optimality Algorithm

First, we wish to put the concurrency constraint in our algorithm. Student groups with concurrency issue are separated from that of 24-hour period conflict. Among the concurrent exams, an exam with the least number of students is to be re-scheduled at other time period. Then the existing exam schedule is removed. Repeating the above process provides us with the adapted schedule without concurrency issue but only with 24-hour period constraint. After imposing concurrency constraint, the number of conflicts was reduced to 31 from 46.

Simple optimality algorithm

In dealing with the second and the most important constraint, 24-hour period constraint, we used what we named as simple optimally algorithm. With an initial schedule given from the result using Java, we apply simple optimal algorithm to get optimal final exam schedule. First, we have to check at which time periods students have three consecutive exams within 24-hour period by using these constraints:

$$X_{i,g,k} = \begin{cases} 1 & \sum_{t=k}^{t=k+2} m_{i,j,g} = 3, k \in [1, 16] \\ 0 & \text{else} \end{cases}$$

$$m_{i,j,g} = \begin{cases} 1 & \text{grade level g students majoring in i take exam j at certain } t - \text{period} \\ 0 & \text{else} \end{cases}$$

$$i \in [1, 23], j \in [1, 126], g \in [1,4], t \in [1, 18]$$

If the sum of $m_{i,j,g}$ in three consecutive periods equals to 3, then grade level-g students majoring in i have three exams within 24 hours. In this schedule, we find that 150 CS major sophomores from time period 2 to 4 and 50 philosophy major sophomores from time period 14 to 16 have three consecutive exams. Refer to Appendix C (a)

As our objective is to minimize the number of students who takes three exams consecutively, we decide to move the second exam to another period, before the first exam or after the last one. In order to choose the value of t, we should consider two things: conflicts and room capacity. Using the constraints below, we can find a better period for the second exam:

$$m_{i,j,g} = \begin{cases} 1 & \text{grade level g students majoring in i take exam j at certain t} - \text{period} \\ 0 & \text{else} \end{cases}$$

$i \in [1, 23]$, $j \in [1, 126]$, $g \in [1,4]$

If $\sum_j m_{i,j,g} = 1$ or 0 at given t, there is no conflict.
If $\sum_j m_{i,j,g} > 1$, there is a conflict.

$S_{j,g,t}$ = # of grade level g-students taking exam j at t

$j \in [1, 126]$, $t \in [1, 18]$

$$r_{l,t} = \begin{cases} 1 & \text{room l is not used yet} \\ 0 & \text{else} \end{cases}$$

$l \in [1, 35]$, $t \in [1, 18]$

n_l = capacity of room l / 2 (every other seat)

$$\sum_g S_{j,g,t} \leq \sum_l n_l \cdot r_{l,t}$$

The second exam of CS sophomores, 15-212 principle of programming course, can be moved to t <2, or t > 4. Since it has conflicts with 85-102 at time period 6, and 21-241 at time period 14, we can exclude t = 6, and 14. Even though exam of course number 15-212 does not have a conflict at time period 5, we cannot choose t=5 because CS sophomores would have another three consecutive exams from time period 4 to 6. Among the rest of t-values, for enough room capacity, we take the period with the smallest total number of students taking exams. At time period 9, 15, and 18, total 660 students, the smallest number, have exams. Finally, we compare remaining room capacity: at time period 18, the largest room capacity exists. Therefore, for the best schedule, we should move 15-212 course exam to the third period on the last day. Similarly, the second exam of philosophy major sophomores, 21-256 Multivariate Analysis and Approximations course, can be moved to t<14, or t>16. We can exclude t=1, 12, 13 because of conflicts with 15-110 and 36-225, and another three consecutive exams from time period 12 to 14. After moving 15-212 course exam, time period 3 has the smallest total number of students, 480 students. Since there is enough room capacity at t=3, we should move 21-256 course exam to third period on the first day.

RESULT

Since there are no students who are taking three exams in 24-hour period, the schedule in Appendix C (b) is one of the optimal final exam schedules that we could get through simple optimal algorithm.

IN REALITY

Carnegie Mellon University has been continuously dealing with the issues on final exam schedule every semester. Due to the large number of exams and students, it is evidently difficult to construct a schedule that fits for every student on campus. According to the CMU registrar, they have been using software called "Schedule Expert" from Strathman Associates. At the end of the add/drop period, they download each student course registration and the courses with exams and cross-listed courses, and then run the program to create a schedule, which minimizes direct conflicts and three exams within 24-hour period. CMU pre-schedules large courses, first year student courses, or special faculty request with little difficulty when it makes sense to grant the request.

FURTHER

In conclusion, to our surprise, we have come up with *the* optimal final exam schedule modeled upon 2010 Fall Semester CMU. We believe our optimal schedule was obtained purely from the benefit of simplistic assumptions. We dealt only with limited number of classes and majors, and restricted students' freedom to take courses. In reality, each student manages to create their own schedule based on their personal need and interest. About half the students have minor or other major, some re-take courses for academic or personal reasons and some have cross-department majors that has its unique catalog to follow.

In simple word, reality is far more complex. There are far too many variables, combinations, possibilities, and exceptions to consider. In pursuit of achieving a more insightful and competitive algorithm to take further steps, we have come up with a rebalancing algorithm.

The rebalancing algorithm is the following: it is an algorithm that balances or obtains the optimal schedule, given an extra schedule to add in or extract out from the existing feasible schedule. The basic thought being - by iteratively adding a new exam to our current optimal exam schedule, we would eventually attain realistic final schedule. This way, we can extend our scope of insight from a feasible solution to a *better* solution.

Yet, noting the difficulty and intricacy of the problem, we faced an invincible challenges that are far beyond of our scope of knowledge both in programming-wise and approach-wise. Hence, we would like to let prospective future students who may want to research similar problem to take above suggestive approach into consideration and possibly get better optimal schedule.

Appendix

APPENDIX A

Undergraduate First-Year Admission Statistics for 2010-2011 Year:

Undergraduate First-Year Admission Statistics 2010-2011

Enrollment as of September 14, 2010

College	Applied	Admitted	Enrolled	SAT-CR[1]	SAT-M[1]
CIT	7,046	1,895	434	630-720	710-790
CFA	3,598	675	260	600-690	610-710
H&SS	3,991	1,201	261	620-710	650-740
IS	800	159	53	600-700	670-750
MCS	5,059	1,467	228	630-730	710-790
SCS	3,046	434	143	670-750	750-800
Tepper	3,082	465	83	610-670	700-790
BHA/BSA/BCSA[3]	96	96	24	710-770	690-790
Total	**26,718**	**6,392**	**1,486**		

Appendix A

Assumed number of students for each major in each department enrolled for fall in 2010:

Tepper 100 1) Business 50 2) Econ 50 **MCS 250** 1) Biology 60 2) Chem 60 3) Physics 60 4) Math 70 **CIT 450** 1) ECE 90 2) MechE 90 3) ChemE 90 4) CivilE 90 5) MaterialScienceE 90	**H&SS 250** 1) Psych 50 2) English 50 3) Stat 50 4) Philosophy 50 5) Social&Decision Sciences 50 **CFA 250** 1) Fine Art 60 2) Design 60 3) Archi 60 4) Music 40 5) Drama 30 **SCS 150** **IS 50**

→ **Total number of students per grade level: 1500**

APPENDIX A

Sample Survey for Elective Requirements:

Grade: Freshmen / Sophomore / Junior / Senior

Department:

Major:

- What elective courses are you taking this semester?

    1)

    2)

    3)

- What elective courses are you planning to take later / have you taken before?

    1)

    2)

    3)

APPENDIX A

Sample Suggested Schedule from CMU catalogue:

Suggested Schedule

Freshman Year

	Fall	Units
21-120	Differential and Integral Calculus	10
33-111	Physics for Science Students I	12
15-100	Introductory/Intermediate Programming	10
03-121	Modern Biology	9
76-101	Interpretation and Argument	9
99-101	Computing @ Carnegie Mellon	3

Sophomore Year

	Fall	Units
21-228	Discrete Mathematics (or 21-484)	9
21-241	Matrix Algebra	9
21-259	Calculus in Three Dimensions	9
21-201	Undergraduate Colloquium	1
73-100	Principles of Economics	9

Junior Year

	Fall	Units
21-369	Numerical Methods	9
xx-xxx	Depth Elective	9
36-225	Introduction to Probability and Statistics I (or 21-325)	9
73-150	Microeconomics	9
xx-xxx	Elective	9

Senior Year

	Fall	Units
21-393	Operations Research II	9
xx-xxx	Depth Elective	9
36-401	Modern Regression	9
xx-xxx	H&SS Elective	9
xx-xxx	Elective	9
		45

APPENDIX A

Assumed Students' Course Schedules for fall semester in 2010:

Carnegie Institute of Technology (CIT)

1) Electrical and Computer Engineering

First Year:
21-122 Integration, Differential Equations and Approximation
76-101 Interpretation and Argument
99-101 Computing at Carnegie Mellon
15-110 Principles of Computing
82-131 Elementary Chinese I

Second Year:
33-106 Physics for Engineering Students I
18-220 Electronic Devices and Analog Circuits
18-231 Sophomore Projects
18-240 Structure and Design of Digital Systems
18-290 Signals and Systems

Third Year:
18-320 Microelectronic Circuits
18-342 Fundamentals of Embedded Systems
18-370 Fundamentals of Control
21-259 Calculus in Three Dimensions

Fourth Year;
18-431 Undergraduate Projects – Senior
18-450 Digital Wireless Communications
18-491 Fundamentals of Signal Processing
18-493 Electroacoustics

2) Mechanical Engineering

First Year:
21-122 Integration, Differential Equations and Approximation
76-101 Interpretation and Argument
99-101 Computing at Carnegie Mellon
15-110 Principles of Computing
82-131 Elementary Chinese I

Second Year:
33-106 Physics for Engineering Students I
21-259 Calculus in Three Dimensions
42-221 Thermodynamics I
24-261 Statics

Third Year:
24-302 Mechanical Engineering Seminar
24-311 Numerical Methods
24-322 Heat Transfer
24-351 Dynamics
36-220 Engineering Statistics and Quality Control

Fourth Year:
24-441 Design II: Conceptualization and Realization
24-452 Mechanical Systems Experimentation
24-424 Energy and the Environment
24-451 Feedback Control Systems

3) Chemical Engineering

First Year:
21-122 Integration, Differential Equations and Approximation
76-101 Interpretation and Argument
99-101 Computing at Carnegie Mellon
42-101 Introduction to Biomedical Engineering
09-105 Introduction to Modern Chemistry I

Second Year:
09-106 Modern Chemistry II
15-110 Principles of Computing
82-131 Elementary Chinese I
06-222 Sophomore Chemical Engineering Seminar
06-221 Thermodynamics

Third Year:
06-323 Heat and Mass Transfer
06-321 Chemical Engineering Thermodynamics
09-347 Advanced Physical Chemistry
09-217 Organic Chemistry I
82-231 Intermediate Chinese I

Fourth Year:
06-421 Chemical Process Systems Design
06-422 Chemical Reaction Engineering
06-423 Unit Operations Laboratory
42-401 Foundation of BME Design

4) Biomedical Engineering

First Year:
21-122 Integration, Differential Equations and Approximation
76-101 Interpretation and Argument
99-101 Computing at Carnegie Mellon
42-101 Introduction to Biomedical Engineering
03-121 Modern Biology

Second Year:
15-110 Principles of Computing
82-131 Elementary Chinese I
42-203 Biomedical Engineering Laboratory
42-201 Professional Issues in Biomedical Engineering
06-221 Thermodynamics

Third Year:
06-323 Heat and Mass Transfer
42-300 Junior BME Research Project
42-341 Introduction to Biomechanics
09-347 Advanced Physical Chemistry

Fourth Year
06-421 Chemical Process Systems Design
06-423 Unit Operations Laboratory
42-444 Medical Devices
42-401 Foundation of BME Design

5) Civil and Environmental Engineering

First Year:
21-122 Integration, Differential Equations and Approximation
76-101 Interpretation and Argument
99-101 Computing at Carnegie Mellon
33-106 Physics for Engineering Students I
82-131 Elementary Chinese I

Second Year:
12-212 Statics
21-259 Calculus in Three Dimensions
09-101 Intro to Experimental Chemistry
09-105 Modern Chemistry I

Third Year:
21-301 Civil and Environmental Engineering Projects
12-335 Soil Mechanics
12-336 Soil Mechanics Lab
12-355 Fluid Mechanics
12-356 Fluid Mechanics Lab

Fourth Year:
12-401 Civil and Environmental Engineering Design
12-411 Project management
12-421 Engineering Economics
12-600 Auto CAD
12-651 Air Quality Engineering

College of Humanities and Social Sciences (H&SS)

1) Psychology

First Year:
76-101 Interpretation and Argument
76-145 Freshman Seminar
21-111 Calculus I
36-201 Statistical Reasoning and Practice
03-121 Modern Biology

Second Year:
21-120 Differential and Integral Calculus
85-211Cognitive science
79-104 World History
33-124 Astronomy
85-241 Social Psychology

Third Year:
85-219 Biological Foundations of Behavior
09-105 Introduction to Modern Chemistry I
15-102 Exploring Programming with Graphics
33-111 Physics I for Science Students

Fourth Year
85-370 Perception
85-421 Language and Thought
85-310 Research Methods in Cognitive Psychology
36-309 Experimental Design for Behavioral and Social Sciences

2) English

First Year:
76-101 Interpretation and Argument
76-145 Freshman Seminar
36-201 Statistical Reasoning and Practice
79-104 Global Histories
21-120 Differential and Integral Calculus

Second Year:
76-272 Language in Design
76-270 Writing for the Professions
76-239 Introduction to Film Studies
76-247 Shakespeare: Comedies and Romances
76-321 Genre Studies

Third Year:
76-260 Survey of Forms: Fiction
76-294 Interpretive Practices
76-394 Research in English Studies
76-386 Language & Culture
76-390 Style

Fourth Year:
76-425 Science in the Public Sphere
76-397 Instructional Text Design
76-462 Advanced Fiction Workshop
76-391 Document Design
76-306 Editing and Publishing

3) Statistics
First Year:
21-120 Differential and Integral Calculus
36-201 Statistical Reasoning and Practice
76-101 Interpretation and Argument
76-145 Freshman Seminar
79-104 World History

Second Year:
21-256 Multivariate Analysis and Approximations
21-241 Matrix Algebra
36-225 Introduction to Probability and Statistics I
15-110 Principles of Computing

Third Year:
73-150 Microeconomics
36-350 Data Mining
21-127 Concepts of mathematics
21-292 Operations Research I

Fourth Year:
36-401 Modern Regression
36-463 Topics in Statistics
80-226 Revolutions in Science
88-305 Rational Choice

4) Philosophy

First Year:
21-120 Differential and Integral Calculus
36-201 Statistical Reasoning and Practice
76-101 Interpretation and Argument
76-145 Freshman Seminar
79-104 World History

Second Year:
21-256 Multivariate Analysis and Approximations
21-241 Matrix Algebra
80-226 Revolutions in Science
80-130 Introduction to Ethics
15-110 Principles of Computing

Third Year:
80-208 Critical Thinking
80-220 Logic and Proofs
80-365 Ramsey
80-210 Logic and Proofs

Fourth Year:
80-383 Language in Use
80-270 Philosophy of Mind
80-226 Revolutions in Science
80-413 Category Theory

5) Social & Decision Science

First Year:
21-111 Calculus I
36-201 Statistical Reasoning and Practice
76-101 Interpretation and Argument
76-145 Freshman Seminar
79-104 World History

Second Year:
21-120 Differential and Integral Calculus
88-122 Introduction to Game Theory & Strategy
85-221 Cognitive Psychology
15-110 Principles of Computing

Third Year:
88-220 Policy Analysis I
88-223 Decision Analysis and Decision Support Systems
88-302 Behavioral Decision Making
88-377 Attitude Persuasion

Fourth Year:
70-381 Marketing
88-412 Economics of Global Warming
73-347 Game Theory for Economists
88-387 Social Norms and Economics

College of Fine Arts (CFA)

1) Architecture

First Year:
48-100 Architecture Design Studio: Foundation I
48-120 Introduction to Digital Media I
48-130 Architectural Drawing I: A Tactile Foundation
21-114 Calculus for Architecture (mini 2)
64-100 Critical Histories of the Arts

Second Year:
48-200 Architecture Design Studio: Composition
48-210 Statics
48-230 Architectural Drawing III: Perspective
48-240 Architecture History I: Historical Survey of World Architecture and Urbanism
76-101 Interpretation and Argument

Third Year:
48-300 Architecture Design Studio: Site
48-312 Site Engineering and Foundations
48-315 Environment I: Climate and Energy
48-448 History of Sustainable Architecture
99-101 Computing @ Carnegie Mellon

Fourth Year:
48-400 Architecture Design Studio: Occupancy
48-410 Environment II: Acoustics and Light
48-412 Environment III: Mechanical Systems
48-452 Real Estate Design and Development
48-338 European Cities in the XIX Century: Planning, Architecture, Preservation

Fifth Year:
48-500 Architecture Design Studio: The Urban Lab
48-497 Thesis Preparation (optional)
48-550 Issues of Practice
48-453 Urban Design Theory and Practice
48-470 Experimenting with Lamination, Clamping, and Cutting

2) Fine Arts

First Year
60-101 Concept Studio I
60-110 Electronic Media Studio I: Computer Art
60-150 2D Media Studio I: Drawing
60-104 Contemporary Issues Forum
99-101 Computing @ Carnegie Mellon
76-101 Interpretation and Argument

Second Year
60-201 Concept Studio III 10
60-230 3D Media Studio II: Foundry, Metals, Construction 10
60-250 2D Media Studio III: Painting 10
60-205 Modern Visual Culture: 1789-1945
79-104 Global Histories

Third Year
60-301 Contextual Practice
60-409 Advanced ETB: Video & Performance
60-415 Advanced ETB: Animation
60-486 The Art and Science of Color
82-131 Chinese

Fourth Year
60-401 Senior Project
60-435 SIS:Metals
60-432 Advanced CP/SIS Site-Work Braddock
60-371 Breathelss: Internation new Wave Cinemas
73-100 Principles of Economics

3) Design

First Year
51-101 Design Studio I
51-121 Design Drawing I
51-171 Human Experience in Design
76-101 Interpretation & Argument
85-100 Introduction to Intelligence
99-101 Computing @ Carnegie Mellon

Second Year
51-201 Basic Typography: CD Studio I
51-203 Communication Design Computer Lab
51-241 How People Work: Human Factors
51-229 Digital Photographic Imaging
51-271 Design History I
51-785 Designing for Service

Third Year
51-301 Advanced Typography: CD Studio III
51-321 Photography and Communication
51-327 Web Design
51-399 Junior Independent Study
51-765 Introduction to Industrial Design Fundamentals
82-131 Elementary Chinese I

Fourth Year
51-403 Sr. Project: Interaction Design
51-421 Visual Interface Design
51-471 Issues of Professional Practice
51-499 Senior Independent Study
51-707 Visual Processes
82-132 Elementary Chinese

4) School of Drama

First year
54-011 Warmup
54-101 Acting I
54-103 Speech I
54-105 Voice/Alexander I
54-107 Movement I
54-111 Text
54-159 Production Symposium I
54-175 Conservatory Hour
54-177 Foundations of Drama I
64-100 Critical Histories of the Arts

Second year
54-201 Acting II
54-203 Voice & Speech II
54-207 Movement II
54-211 Actor Dance II
54-221 Directing II
54-259 Production Symposium II
54-281 Foundations of Drama III
82-1311 Elementary Chinese I

Junior Year
54-301 Acting III 12
54-303 Speech III (accents) 6
54-305 Voice/Alexander III 6
54-307 Movement III 6
54-311 Acting Symposium III 10
54-325 Actor Dance III 3
54-382 History of Drama
82-131 Elementary Chiense II

Senior Year
54-401 Camera Lab
54-405 Graduate Directing
54-407 Movement IV
54-409 Theatre Lab
54-411 Acting Symposium IV
54-413 Showcase
54-437 Acting for the Camera
54-494 Business of Acting
21-120 Differential and Integral Calculus

5) School of Music

First year:
57-501 Studio
57-420 Jazz Vocal Ensemble
57-193 Skills of Accompanying I
57-152 Harmony I
57-161 Eurhythmics I
57-181 Solfege I 3 57-189 Repertoire and Listening for Musicians I
57-173 Survey of Western Music History
76-101 Interpretation and Argument

Second Year
57-501 Studio
57-417 Major Choral Ensemble
57-228 Chamber Music
57-151 Principles of Counterpoint
57-163 Eurhythmics III
57-183 Solfege III
57-289 Repertoire and Listening for Musicians III
57-283 Music History I
82-171 Elem. Japanese

Third Year
57-501 Studio
57-418 Major instrumental ensemble
57-228 Chamber Music
57-480 History of Black American Music
82-271 Inter. Japanese
57-103 Elective Studio(Beginning Piano Class)

Fourth Year
57-501 Studio
57-234 Performance for Composers
57-349 Supervised Theory Teaching
82-141 Elementary Spanish
57-107 Elective Studio(Jazz Piano I class)

The Tepper School of Business (Tepper)

1) Undergraduate Business Administration

First year:
73-100 Principles of Economics
70-100 Introduction to Business
21-120 Differential and Integral Calculus
76101 Interpretation and Argument
99-101 Computing at CMU

Second year:
21-257 Models and Methods of Optimization
70-122 Introduction to Accounting
70-207 Probability and Statistics
79-104 Introduction to World History

Third year:
70-371 Production and Operations management
70-381 Marketing
73-200 Macroeconomics
70-391 Finance
70-451 Management Information Systems

Fourth year:
70-201 Service project
85-102 Introduction to Psychology
70-492 Investment Analysis
70-440 Corporate Strategy

2) Undergraduate Economics Program

First year:
76-101 Interpretation and Argument
15-100 Introductory Programming
21-120 Differential and Integral Calculus
73-100 Principles of Economics
36-201 Statistical Reasoning

Second year:
36-303 Sampling, Survey and Society
73-270 Writing for Economics
73-310 History of Economic Issues and Analysis
73-200 Intermediate Macroeconomics
82-132 Elementary Chinese 1

Third year:
73-261 Econometrics
76-270 Writing in the Professions
73-252 Advanced Microeconomic Theory
73-253 Advanced Macroeconomic Theory
73-359 Benefit-cost Analyses

Fourth year:
73-347 Game Theory Economics
73-497 Senior Project
76-271 Introduction to Professional and Technical Writing
36-303 Sampling, Survey, and Society

Mellon College of Science (MCS)

1) Department of Mathematical Sciences

First year:
21-120 Differential and Integral Calculus
33-111 Physics 1 for Science
15-100 Introductory Programming
03-121 Modern Biology
76-101 Interpretation and Argument
99-101 Computing at CMU

Second year:
21-228 Discrete Mathematics
21-341 Linear Algebra 1
21-259 Calculus is Three Dimensions
79-104 Introduction to World History
21-201 Undergraduate Colloquium

Third year:
21-355 Principles of Real Analysis 1
36-225 Introduction to Probability and Statistics 1
82-132 Elementary Chinese 1
21-369 Numerical Methods
21-370 Discrete Time Finance

Fourth year:
21-393 Operations Research 2
36-401 Modern Regression
21-600 Math Logic 1
21-476 Ordinary Differential Equations

2) Department of Chemistry

First year:
09-105 Introduction to Modern Chemistry
21-120 Differential and Integral Calculus
33-111 Physics I for Science
76-101 Interpretation and Argument
99-101 Computing at CMU

Second year:
09-201 Undergraduate Seminar 1
09-219 Modern Organic Chemistry 1
09-221 Lab 1: Introduction to chemical Analysis
09-231 Mathematical Methods for Chemists
03-121 Modern Biology

Third year:
09-301 Undergraduate Seminar 3
09-321 Lab 3: Molecular Design and Synthesis
09-344 Physical Chemistry
09-331 Modern Analytical Instrumentation
09-507 Nanoparticles

Fourth year:
09-401 Undergraduate Seminar 5
09-445 Undergraduate Research
09-711 Physical Organic Chemistry
09-518 Bioorganic Chemistry

3) Department of Physics

First year:
33-111 Physics 1 for Science Students
15-100 Introductory Programming
21-120 Differential and Integral Calculus
99-101 Computing at CMU
76-101 Interpretation and Argument

Second year:
33-211 Physics 3: Modern Essentials
33-231 Physical Analysis
21-259 Calculus in Three Dimensions
09-105 Introduction to Modern Chemistry
33-201 Physics Sophomore Colloquium 1

Third year:
33-331 Physical Mechanics 1
33-338 Intermediate Electricity and Magnetism 1
33-341 Thermal Physics 1
33-301 Physics Upper Class Colloquium
82-132 Elementary Chinese 1

Fourth year:
21-228 Discrete Mathematics
33-445 Advanced Quantum Physics 1
79-104 Introduction to World History
85-102 Introduction to Psychology

4) Department of Biological Sciences

First year:
03-121 Modern Biology
09-105 Introduction to Chemistry
21-120 Differential and Integral Calculus
99-101 Computing at CMU
76-101 Interpretation and Argument

Second year:
03-201 Undergraduate Colloquium Sophomore
03-231 Biochemistry
21-259 Calculus in Three Dimensions
33-111 Physics 1 for Science Students
03-240 Cell Biology

Third year:
03-301 Undergraduate Colloquium Junior
03-343 Experimental Genetics and Molecular Biology
03-325 Evolution
82-171 Elementary Japanese 1

Fourth year:
03411 Topics in Research
79-104 Introduction to World History
03-330 Genetics
03-401 Undergraduate Colloquium Seniors

Information Systems (IS)

First year:
76-101 Interpretation and Argument
36-201 Statistical Reasoning
21-120 Differential and Integral Calculus
15-100 Introductory Programming
99-101 Computing at CMU

Second year:
67-250 Information Systems Milieux
36-201 Statistics Reasoning Practics
88-220 Policy Analysis 1
57-337 Sound Recording
79-104 Introduction to World History

Third year:
67-271 Fundamentals of Systems Development
67-306 Management Computers Information Systems
67-371 Fundamentals System Development 1
70-414 Tech Based Entre

Fourth year:
67-475 Information Systems Applications
85-102 Introduction to Psychology

School of Computer Science (SCS)

First year:
15-110 Intermediate/Advanced Programming
15-128 Freshman Immigration Course
21-120 Differential & Integral Calculus
21-127 Concepts of Mathematics
76-101 Interpretation and Argument
99-101 Computing Skills Workshop
09-105 Introduction to Modern Chemistry

Second year:
15-123 Effective Programming in C and UNIX
15-212 Principles of Programming
21-241 Matrix Algebra
03-121 Biology
85-102 Introduction to Psychology

Third year:
15-451 Algorithm Design and Analysis
15-410 Operating System Design and Implementation
36-225 Introduction to Probability and Statistics I
73-100 Principles of Economics
70-381Marketing I

Fourth year:
15381 Artificial Intelligence: Representation and Problem Solving
82-273 Introduction to Japanese Language and Culture
70-483 Advertising and Marketing Communications
70-122 Introduction to Accounting

APPENDIX A

Classroom Information where the Final Exams are taken:

Room Number	Capacity	Room Number	Capacity
BH 235A	35	MM A14	110
BH 235B	35	PH 100	234
BH 237B	35	PH 125C	70
BH 255A	35	PH A18A	50
DH 1112	99	PH A18B	50
DH 1117	30	PH A18C	50
DH 1211	35	SH 124	95
DH 1212	107	SH 214	45
DH 1217	35	SH 219	45
DH 2210	278	UC McConomy	445
DH 2315	258	WEH 5304	20
DH A310	60	WEH 5320	30
GHC 4215	53	WEH 6423	35
GHC 4301	28	WEH 8427	35
GHC 4307	75	WEH 5403	65
HH B 103	96	WEH 5302	40
HH B 131	96	Hamberg Hall 1000	126
MM 103	115		
		Total Capacity	3050

APPENDIX B

```
import java.util.*;


public class Course
{
  public int courseNumber;
  public int studentNumber;
  public int numLeft;
  public List<String> room;
  public int time;

  public Course(int courseNumber, int studentNumber) {
    this.courseNumber = courseNumber;
    this.studentNumber = studentNumber;
    this.numLeft = studentNumber;
    this.room = new ArrayList<String>();
  }

  public void courseStatus() {
    System.out.print("Course " + this.courseNumber);
    if (room != null)
      System.out.println(" has been assigned to room " + room + " at time " + time);
    else
      System.out.println(" has no assigned room.");
  }

  public boolean fillThis(Room[] rooms, int time) {
    // checks whether there are enough spaces in this time to fit in the schedule
    int totalSpaceInTime = totalSpaceInTime(rooms,time);
    if (totalSpaceInTime/2 < this.studentNumber) {
      return false;
    }
    else {
      for (int i=1;i<rooms.length;i++) {
        if (numLeft != 0 && rooms[i].spaceAtTime(time) !=0) {
          this.fillIn(rooms[i],time);
          this.room.add(rooms[i].roomName());
          this.time = time;
        }
      }
      return true;
    }
  }
  public boolean fill(Room[] rooms, int time) {
```

```
    if (time == 19)
      return false;
    // checks whether there are enough spaces in this time to fit in the schedule
    int totalSpaceInTime = totalSpaceInTime(rooms,time);
    if (totalSpaceInTime/2 < this.studentNumber) {
      // go with the next time period
      return fill(rooms,time+1);
    }
    else {
      for (int i=1;i<rooms.length;i++) {
        if (numLeft != 0 && rooms[i].spaceAtTime(time) !=0) {
          this.fillIn(rooms[i],time);
          this.room.add(rooms[i].roomName());
          this.time = time;
        }
      }
      return true;
    }
  }
  private void fillIn(Room room, int time) {
    room.scheduleAtTime(this, time);
  }
  private int totalSpaceInTime(Room[] rooms, int time) {
    int totalSpace = 0;
    for (int i=1; i<rooms.length; i++) {
      totalSpace += rooms[i].spaceAtTime(time);
    }
    return totalSpace;
  }

  public String toString() {
    String t = "Course: " + courseNumber + ", " + studentNumber + " students";
    return t;
  }
}

public class Room
{
  // create an array of Room with roomNumber, time(18 for each), capacity and availability
  public String room;
  public Node root;
  public int capacity;

  public Room(String room, int capacity) {
    this.room = room;
    this.capacity = capacity;
```

```
}

public void removeCourse(int time) {
  Node temp = this.root;
  while (temp != null && temp.time != time) {
    temp = temp.next;
  }
  if (temp.course != null)
    temp.course = null;
  temp.space = this.capacity;
}

public String roomName() {
  String t = this.room;
  return t;
}

public void createTime() {
  root = new Node(18,this.capacity);
  for (int i=17;i>0;i--)
    root.add(new Node(i,this.capacity));
}

public int spaceAtTime(int time) {
  Node temp = this.root;
  while (temp != null) {
    if (temp.time == time && temp.isOpen()) {
      return temp.space;
    }
    temp = temp.next;
  }
  return 0;
}

public int scheduleAtTime(Course course, int time) {
  Node temp = this.root;
  while (temp != null && temp.time != time) {
    temp = temp.next;
  }
  int toReturn = 0;
  if (course.numLeft - temp.space/2 >= 0)
    toReturn = temp.space;
  else
    toReturn = course.numLeft;
  temp.space = 0;
  temp.course = course;
```

```
  course.numLeft -= toReturn;
  return toReturn;
}


public void roomStatus() {
  Node temp = this.root;
  System.out.println("Room " + this.room + " has the following exam schedule: ");
  while (temp != null) {
    System.out.print("Time: " + temp.time + ". Status: ");
    if (temp.isOpen())
      System.out.println("Open");
    else
      System.out.println("Filled with course " + temp.course.courseNumber);
    temp = temp.next;
  }
}

public String toString() {
  String t = "Room: " + room + " Capacity: " + capacity;
  return t;
}

public boolean isOpen(int time) {
  Node temp = this.root;
  while (time>1) {
    temp = temp.next;
    time--;
  }
  return temp.isOpen();
}

//availability of room in each time period represented as a linked list
public class Node {
  int time;
  Node next;
  Course course;
  int space;

  public Node(int time, int space) {
    this.time = time;
    next = null;
    this.space = space;
  }

  public boolean isOpen() { return space != 0; }
```

```
    public void add(Node t) {
      Node temp = root;
      root = new Node(t.time,t.space);
      root.next = temp;
    }
  }
}

import java.util.*;

public class Student
{
  public String major;
  public int grade;
  public int number;
  public List<Integer> courseList;

  public Student(String major, int grade, int number, int[] course)
  {
    this.major = major;
    this.grade = grade;
    this.number = number;
    this.courseList = new ArrayList<Integer>();
    for (int i=0; i<course.length; i++) {
      this.courseList.add(course[i]);
    }
  }

  public String toString() {
    String year;
    if (grade == 1)
      year = "Freshman";
    else if (grade == 2)
      year = "Sophomore";
    else if (grade == 3)
      year = "Junior";
    else
      year = "Senior";
;    String t = year + " " + major + " major";
    return t;
  }
}

import java.util.*;
```

```
public class ORFinal {
  public static void main(String[] args) {
    Student[] students = fall10Student();
    List<Course> courses = fall10Course();
    Room[] rooms = fall10Room();

    initialSchedule(courses,rooms);
    List<Student> unfits = new ArrayList<Student>();
    int conflicts = conflictStatus(courses,students,unfits);

    ///////////////  Things to show ////////////////////

    //              ROOM STATUS              //
    //System.out.println("ROOM STATUS : ");
    //for (int i=1; i<rooms.length; i++) {
      //rooms[i].roomStatus();
      //System.out.println();
    //}

    //              COURSE STATUS              //
    //System.out.println("COURSE STATUS : ");
    //for (int i=0;i<courses.size();i++)
      //courses.get(i).courseStatus();

    //              CONFLICT STATUS              //
     //System.out.println(unfits.toString());

    ///////////////  Things to show ////////////////////

    int counter = 0;
    while (conflicts != 0 && counter != 50) {
      optimizeSchedule(rooms, courses,unfits); // so that no exams are scheduled at the same
period for each students
      unfits = new ArrayList<Student>();
      conflictStatus(courses,students,unfits);
      counter++;
    }
    for (int i=1; i<rooms.length; i++) {
      rooms[i].roomStatus();
      System.out.println();
    }

    //              COURSE STATUS              //
    System.out.println("COURSE STATUS : ");
    for (int i=0;i<courses.size();i++)
      courses.get(i).courseStatus();
```

```
  }

  public static void optimizeSchedule(Room[] rooms, List<Course> courses, List<Student>
unfits) {
    // moving two exams scheduled at the same period - moving the one with less students to other
period
    for (int i=0; i<unfits.size();i++) {
      List<Integer> list = timeList(courses,unfits.get(i).courseList);
      for (int j=0;j<list.size();j++) {
        for (int k=0; k<list.size();k++) {
          if (j != k && list.get(j) == list.get(k)) {
            if (getStudentNumber(courses, unfits.get(i).courseList.get(j)) <
getStudentNumber(courses, unfits.get(i).courseList.get(k)))
              move(rooms, courses, getCourse(courses, unfits.get(i).courseList.get(j)),list.get(j));
            else
              move(rooms,courses,getCourse(courses, unfits.get(i).courseList.get(k)),list.get(k));
          }
        }
      }
    }
  }
  public static void move(Room[] rooms, List<Course> courses, Course course, int from) {
    boolean flag = false;
    for (int i=1;i<19;i++) {
      if (from == 0) {
        course.fill(rooms,1);
        i = 20;
      }
      else if (i != from && flag == false) {
        remove(rooms, course,from);
        flag = course.fillThis(rooms,i);
      }
    }
    if (!flag)
      System.out.println("Cannot fit in "  + course.toString());
  }
  public static void remove(Room[] rooms, Course course,int from) {
    for (int j=0; j<course.room.size();j++) {
      for (int i=1;i<rooms.length;i++) {
        if (rooms[i].room.equals(course.room.get(j)))
          rooms[i].removeCourse(from);
      }
    }
    course.room = new ArrayList<String>();
    course.time = 0;
    course.numLeft = course.studentNumber;
```

```
}
private static Course getCourse(List<Course> courses, int courseNumber) {
  Course course = null;
  for (int i=0; i<courses.size(); i++)
    if (courses.get(i).courseNumber == courseNumber) {
      course = courses.get(i);
  }
  return course;
}

public static int conflictStatus(List<Course> courses, Student[] students, List<Student> unfits)
{
  // returns the status of conflict.
  int conflicts=0;
  for (int i=1; i<students.length; i++) {
    List<Integer> list = timeList(courses,students[i].courseList);
    //              CONFLICT STATUS              //
    //System.out.println(students[i].toString() + "'s exam schedule: " + list.toString());
    conflicts += conflictStatus(students[i],list,unfits);
  }
  System.out.println("Total of " + conflicts + " students have a schedule conflict!" );
  return conflicts;
}
private static int conflictStatus(Student student, List<Integer> list, List<Student> unfits) {
  // Also check that no two exams for a student is scheduled at the same time period
  int counter = 0;
  int counter2 = 0;
  for (int i=0;i<list.size();i++) {
    for (int j=0;j<list.size();j++) {
      if (list.get(i) == list.get(j))
        counter2++;
      if (Math.abs(list.get(i) - list.get(j)) < 4)
        counter++;
    }
  }
  if (counter > 5 + list.size() || counter2 > list.size()) {
    unfits.add(student);
    return 1;
  }
  return 0;
}
private static List<Integer> timeList(List<Course> courses, List<Integer> courseList) {
  List<Integer> list = new ArrayList<Integer>();
  for (int i=0; i<courseList.size();i++) {
    list.add(findTime(courses,courseList.get(i)));
  }
```

```
  return list;
}
private static int findTime(List<Course> courses, int courseNumber) {
  int time = -1;
  for (int i=0;i<courses.size();i++) {
    if (courses.get(i).courseNumber == courseNumber)
      time = courses.get(i).time;
  }
  if (time == -1) {
    throw new IllegalArgumentException("something wrong");
  }
  return time;
}

public static List<Course> unfitCourses(List<Course> courses) {
  List<Course> unfit = new ArrayList<Course>();
  for (int i=0;i<courses.size();i++) {
    if (courses.get(i).room == null)
      unfit.add(courses.get(i));
  }
  return unfit;
}

public static boolean isFeasibleSchedule(List<Course> courses) {
  for (int i=0;i<courses.size();i++) {
    if (courses.get(i).room == null)
      return false;
  }
  return true;
}

public static void initialSchedule(List<Course> courses, Room[] rooms) {
  // initiate the schedule
  List<Course> unfitList = courses;
  for (int i=1;i<19; i++) {
    for (int j=0; j<unfitList.size(); j++)
      initialSchedule(unfitList.get(j),rooms,i);
    unfitList = unfitCourses(courses);
  }
}

// precondition - course is in descending order
private static void initialSchedule(Course course, Room[] rooms,int time) {
  if (!course.fill(rooms,time)) {
  }
}
```

```
private static int getStudentNumber(List<Course> courses, int courseNumber) {
  int number = 0;
  for (int i=0; i<courses.size(); i++)
    if (courses.get(i).courseNumber == courseNumber)
      number = courses.get(i).studentNumber;
  return number;
}

public static Room[] fall10Room() {
  Room[] rooms = new Room[26];
  rooms[1] = new Room("UCMcConomy",450);
  rooms[2] = new Room("DH2210",280);
  rooms[3] = new Room("DH2315",280);
  rooms[4] = new Room("PH100",250);
  rooms[5] = new Room("HH1000",140);
  rooms[6] = new Room("MM103",120);
  rooms[7] = new Room("MMA14",110);
  rooms[8] = new Room("DH1212",110);
  rooms[9] = new Room("SH124",100);
  rooms[10] = new Room("HHB103",100);
  rooms[11] = new Room("HHB131",100);
  rooms[12] = new Room("DH1112",100);
  rooms[13] = new Room("GHC4307",80);
  rooms[14] = new Room("PH125C",70);
  rooms[15] = new Room("BH235",70);
  rooms[16] = new Room("BH237",70);
  rooms[17] = new Room("BH255",70);
  rooms[18] = new Room("WEH5403",70);
  rooms[19] = new Room("DHA310",60);
  rooms[20] = new Room("GHC4215",60);
  rooms[21] = new Room("PHA18A",60);
  rooms[22] = new Room("PHA18B",60);
  rooms[23] = new Room("PHA18C",60);
  rooms[24] = new Room("SH214",60);
  rooms[25] = new Room("SH219",60);
  for (int i=1; i<26; i++)
    rooms[i].createTime();
  return rooms;
}

public static Student[] fall10Student() {
  Student[] students = new Student[80];
  int[] b1 = {73100,70100,21120};
  students[1] = new Student("Business", 1,50, b1);
  int[] b2 = {21257,70122,70207,79104};
```

```
students[2] = new Student("Business", 2,50, b2);
int[] b3 = {70371,73200,70391};
students[3] = new Student("Business", 3,50, b3);
int[] b4 = {85102,70492,70440};
students[4] = new Student("Business", 4,50, b4);
int[] e1 = {15110,21120,73100,36201};
students[5] = new Student("Economics", 1,50, e1);
int[] e2 = {73270,73310,73200};
students[6] = new Student("Economics", 2,50, e2);
int[] e3 = {76270,73253,73359};
students[7] = new Student("Economics", 3,50, e3);
int[] e4 = {73347,73497};
students[8] = new Student("Ecnomics", 4,50, e4);
int[] bio1 = {03121,9105,21120};
students[9] = new Student("Biology", 1,60, bio1);
int[] bio2 = {03201,03231,21259,33111};
students[10] = new Student("Biology", 2,60, bio2);
int[] bio3 = {03301,03325,82171};
students[11] = new Student("Biology", 3,60, bio3);
int[] bio4 = {79104,03330,03401};
students[12] = new Student("Biology", 4,60, bio4);
int[] ma1 = {21120,33111,15110,03121};
students[13] = new Student("Mathematics", 1,70, ma1);
int[] ma2 = {21228,21341,21259,79104};
students[14] = new Student("Mathematics", 2,70, ma2);
int[] ma3 = {21355,36225,21369,21370};
students[15] = new Student("Mathematics", 3,70, ma3);
int[] ma4 = {21600,21476};
students[16] = new Student("Mathematics", 4,70, ma4);
int[] ch1 = {9105,21120,33111};
students[17] = new Student("Chemistry", 1,60, ch1);
int[] ch2 = {9219,9221,9231,03121};
students[18] = new Student("Chemistry", 2,60, ch2);
int[] ch3 = {9344,9507};
students[19] = new Student("Chemistry", 3,60, ch3);
int[] ch4 = {9711,9518};
students[20] = new Student("Chemistry", 4,60, ch4);
int[] ph1 = {15110,21120,33111};
students[21] = new Student("Physics", 1,60, ph1);
int[] ph2 = {9105,21259,33231,33211};
students[22] = new Student("Physics", 2,60, ph2);
int[] ph3 = {33331,33338,33341,33301};
students[23] = new Student("Physics", 3,60, ph3);
int[] ph4 = {21228,33445,79104,85102};
students[24] = new Student("Physics", 4,60, ph4);
int[] i1 = {36201,21120,15110};
```

```
students[25] = new Student("InformationSystems", 1,50, i1);
int[] i2 = {67250,36201,79104};
students[26] = new Student("InformationSystems", 2,50, i2);
int[] i3 = {67306,67371,70414};
students[27] = new Student("InformationSystems", 3,50, i3);
int[] i4 = {85102};
students[28] = new Student("InformationSystems", 4,50, i4);
int[] cs1 = {15110,21120,21127,9105};
students[29] = new Student("ComputerScience", 1,150, cs1);
int[] cs2 = {15123,15212,21241,03121,85102};
students[30] = new Student("ComputerScience", 2,150, cs2);
int[] cs3 = {15451,15410,36225,73100};
students[31] = new Student("ComputerScience", 3,150, cs3);
int[] cs4 = {15381,70122};
students[32] = new Student("ComputerScience", 4,150, cs4);
int[] ps1 = {21111,36201,03121};
students[33] = new Student("Psychology", 1,50, ps1);
int[] ps2 = {21120,85211,79104,33124,85241};
students[34] = new Student("Psychology", 2,50, ps2);
int[] ps3 = {85219,9105,33111};
students[35] = new Student("Psychology", 3,50, ps3);
int[] ps4 = {85370,85421,36309};
students[36] = new Student("Psychology", 4,50, ps4);
int[] en1 = {36201,79104,21120};
students[37] = new Student("English", 1,50, en1);
int[] en2 = {76270,76321};
students[38] = new Student("English", 2,50, en2);
int[] st1 = {21120,36201,79104};
students[39] = new Student("Statistics", 1,50, st1);
int[] st2 = {21256,21241,36225};
students[40] = new Student("Statistics", 2,50, st2);
int[] st3 = {73150,36350,21127};
students[41] = new Student("Statistics", 3,50, st3);
int[] st4 = {36463,80226};
students[42] = new Student("Statistics", 4,50, st4);
int[] p1 = {21120,36201,79104};
students[43] = new Student("Philosophy", 1,50, p1);
int[] p2 = {21256,21241,80226,15110};
students[44] = new Student("Philosophy", 2,50, p2);
int[] p4 = {80383,80226};
students[45] = new Student("Philosophy", 4,50, p4);
int[] sd1 = {21111,36201,79104};
students[46] = new Student("DecisionScience", 1,50, sd1);
int[] sd2 = {21120,88122,15110};
students[47] = new Student("DecisionScience", 2,50, sd2);
int[] sd3 = {88302,88377};
```

```
students[48] = new Student("DecisionScience", 3,50, sd3);
int[] sd4 = {88412,73347};
students[49] = new Student("DecisionScience", 4,50, sd4);
int[] ece1 = {21122};
students[50] = new Student("ECE", 1,90, ece1);
int[] ece2 = {33106,18220,18240,18290};
students[51] = new Student("ECE", 2,90, ece2);
int[] ece3 = {18320,18342,21259};
students[52] = new Student("ECE", 3,90, ece3);
int[] ece4 = {18450,18491};
students[53] = new Student("ECE", 4,90, ece4);
int[] me1 = {21122};
students[54] = new Student("MechE", 1,90, me1);
int[] me2 = {33106,21259};
students[55] = new Student("MechE", 2,90, me2);
int[] me3 = {24302,24322,24351,36220};
students[56] = new Student("MechE", 3,90, me3);
int[] me4 = {24424,24451};
students[57] = new Student("MechE", 4,90, me4);
int[] ce1 = {21122,42101,9105};
students[58] = new Student("ChemE", 1,90, ce1);
int[] ce2 = {9106,06221};
students[59] = new Student("ChemE", 2,90, ce2);
int[] ce3 = {06323,06321,9217};
students[60] = new Student("ChemE", 3,90, ce3);
int[] ce4 = {06422,06423,42401};
students[61] = new Student("ChemE", 4,90, ce4);
int[] be1 = {21122,42101,03121};
students[62] = new Student("BiomedE", 1,90, be1);
int[] be2 = {15110,06221};
students[63] = new Student("BiomedE", 2,90, be2);
int[] be3 = {06323};
students[64] = new Student("BiomedE", 3,90, be3);
int[] be4 = {42444,06423,42401};
students[65] = new Student("BiomedE", 4,90, be4);
int[] cve1 = {21122,33106};
students[66] = new Student("CivilE", 1,90, cve1);
int[] cve2 = {12212,21259,9105};
students[67] = new Student("CivilE", 2,90, cve2);
int[] cve3 = {12335,12355};
students[68] = new Student("CivilE", 3,90, cve3);
int[] ar2 = {48210,48240};
students[69] = new Student("Architecture", 2,60, ar2);
int[] ar3 = {48315};
students[70] = new Student("Architecture", 3,60, ar3);
int[] ar5 = {48550};
```

```
  students[71] = new Student("Architecture", 5,60, ar5);
  int[] fa2 = {79104};
  students[72] = new Student("FineArts", 2,60, fa2);
  int[] fa4 = {73100};
  students[73] = new Student("FineArts", 4,60, fa4);
  int[] de2 = {51271};
  students[74] = new Student("Design", 2,60, de2);
  int[] d4 = {21120};
  students[75] = new Student("Drama", 4,30, d4);
  int[] m1 = {57152,57173};
  students[76] = new Student("Music", 1,90, m1);
  int[] m2 = {57289,82171};
  students[77] = new Student("Music", 2,90, m2);
  int[] m3 = {57480};
  students[78] = new Student("Music", 3,90, m3);
  int[] m4 = {82141};
  students[79] = new Student("Music", 4,90, m4);
  return students;
}

public static List<Course> fall10Course() {
  List<Course> course = new ArrayList<Course>();
  course.add(new Course(21120,830));
  course.add(new Course(15110,800));
  course.add(new Course(79104,600));
  course.add(new Course(9105,560));
  course.add(new Course(21259,560));
  course.add(new Course(21122,500));
  course.add(new Course(03121,480));
  course.add(new Course(73100,360));
  course.add(new Course(21241,350));
  course.add(new Course(33111,300));
  course.add(new Course(36225,220));
  course.add(new Course(21127,200));
  course.add(new Course(70122,200));
  course.add(new Course(85102,200));
  course.add(new Course(36201,200));
  course.add(new Course(73200,200));
  course.add(new Course(33106,180));
  course.add(new Course(06221,180));
  course.add(new Course(06323,180));
  course.add(new Course(06423,180));
  course.add(new Course(36463,180));
  course.add(new Course(42101,180));
  course.add(new Course(76321,150));
  course.add(new Course(15123,150));
```

```
course.add(new Course(15212,150));
course.add(new Course(15381,150));
course.add(new Course(15410,150));
course.add(new Course(15451,150));
course.add(new Course(21228,130));
course.add(new Course(80383,120));
course.add(new Course(82171,110));
course.add(new Course(73347,100));
course.add(new Course(76270,100));
course.add(new Course(21111,100));
course.add(new Course(21256,100));
course.add(new Course(06321,90));
course.add(new Course(06422,90));
course.add(new Course(9106,90));
course.add(new Course(9217,90));
course.add(new Course(12212,90));
course.add(new Course(12335,90));
course.add(new Course(12355,90));
course.add(new Course(18220,90));
course.add(new Course(18240,90));
course.add(new Course(18290,90));
course.add(new Course(18320,90));
course.add(new Course(18342,90));
course.add(new Course(18450,90));
course.add(new Course(42401,90));
course.add(new Course(18491,90));
course.add(new Course(24302,90));
course.add(new Course(24322,90));
course.add(new Course(24351,90));
course.add(new Course(24424,90));
course.add(new Course(24451,90));
course.add(new Course(82141,90));
course.add(new Course(33445,90));
course.add(new Course(21341,70));
course.add(new Course(21355,70));
course.add(new Course(21369,70));
course.add(new Course(21370,70));
course.add(new Course(21476,70));
course.add(new Course(21600,70));
course.add(new Course(03201,60));
course.add(new Course(03231,60));
course.add(new Course(03301,60));
course.add(new Course(03325,60));
course.add(new Course(42444,60));
course.add(new Course(48210,60));
course.add(new Course(48240,60));
```

```
course.add(new Course(48315,60));
course.add(new Course(48550,60));
course.add(new Course(03330,60));
course.add(new Course(03401,60));
course.add(new Course(9219,60));
course.add(new Course(9221,60));
course.add(new Course(9231,60));
course.add(new Course(9344,60));
course.add(new Course(9507,60));
course.add(new Course(33124,60));
course.add(new Course(33211,60));
course.add(new Course(33231,60));
course.add(new Course(33301,60));
course.add(new Course(33331,60));
course.add(new Course(33338,60));
course.add(new Course(33341,60));
course.add(new Course(9518,60));
course.add(new Course(9711,60));
course.add(new Course(21257,50));
course.add(new Course(70100,50));
course.add(new Course(70207,50));
course.add(new Course(70371,50));
course.add(new Course(70391,50));
course.add(new Course(70440,50));
course.add(new Course(70492,50));
course.add(new Course(73253,50));
course.add(new Course(73270,50));
course.add(new Course(73310,50));
course.add(new Course(73359,50));
course.add(new Course(73497,50));
course.add(new Course(33106,50));
course.add(new Course(36220,50));
course.add(new Course(36225,50));
course.add(new Course(36309,50));
course.add(new Course(36350,50));
course.add(new Course(57480,50));
course.add(new Course(67250,50));
course.add(new Course(67306,50));
course.add(new Course(67371,50));
course.add(new Course(70414,50));
course.add(new Course(73150,50));
course.add(new Course(80226,50));
course.add(new Course(85102,50));
course.add(new Course(85211,50));
course.add(new Course(85219,50));
course.add(new Course(85241,50));
```

```
    course.add(new Course(85370,50));
    course.add(new Course(85421,50));
    course.add(new Course(88122,50));
    course.add(new Course(88302,50));
    course.add(new Course(88377,50));
    course.add(new Course(88412,50));
    course.add(new Course(51271,30));
    course.add(new Course(57152,30));
    course.add(new Course(57173,30));
    course.add(new Course(57289,30));
    return course;
  }
}
```

APPENDIX C

(a)

Per. /Day	Room (Capacity)	1	2	3	4	5	6
1st	UC Mc.(445)	15110(800)	**03121(480)**	21120(830)	21259(460)	79104(600)	09105(560)
	D2210(278)	•	•	•	•	•	•
	DH2315(258)	•	•	•	•	•	•
	PH 100(234)	•	09344(60)	•	42401(180)	•	•
	HH1000(126)	•	03201(60)	•	•	15381(150)	**80226(180)**
	MM 103(115)	•		•	42101(180)	•	•
	MMA14(110)	•		•	•	•	•
	DH1212(107)	15451(150)		•	•	03325(60)	
	DH 1112(99)	•		15410(150)	03301(60)	•	
	HH B103(96)	•	03330(60)	•	•	09219(60)	03401(60)
	HH B131(96)	21600(70)	•	•	09221(60)	•	•
	SH 124(95)	•		03231(60)	•		
	GHC4307(75)	09507(60)		•	57480(30)		
	PH 125C(70)	•					
	WEH5403(65)	57152(30)					
	DH A310(60)		57173(30)				82271(30)
	GHC4215(53)						
	PH A18A(50)			09231(60)		88377(50)	88412(50)
	PH A18B(50)			•		•	•
	PH A18C(50)			•			
	SH 214(45)						
	SH 219(45)						
	WEH5302(40)	15110					
	WEH6423(35)						
	WEH8427(35)						
	BH 235A(35)						
	BH 235B(35)						
	BH 237B(35)						
	BH 255A(35)						
	DH 1211(35)						

	DH 1217(35)						
	WEH5320(30)						
	DH 1117(30)			15410	42101		
	GHC4301(28)						
	WEH5304(20)						80226
2nd	UC Mc.(445)	70122(200)	21122(460)	36225(220)	36201(400)	82141(120)	85102(110)
	DH 2210(278)	**15123(150)**	•	21228(130)	•	**21241(250)**	33106(270)
	DH 2315(258)	•	•	24351(90)	•	•	•
	PH 100(234)	24302(90)	73347(100)	33211(60)	24424(90)	73100(100)	36220(90)
	HH 1000(126)	21476(70)	24322(90)	33331(60)	33231(60)	24451(90)	•
	MM 103(115)	21369(70)	•	88302(50)	85211(50)	•	42444(90)
	MM A14(110)	•	33111(300)		88122(50)	85421(50)	•
	DH 1212(107)	09518(60)	•		21355(70)	21341(70)	85241(50)
	DH 1112(99)	•	•		•	•	
	HH B103(96)	21476		48210(60)	33301(60)	33341(60)	33338(60)
	HH B131(96)	48315(60)	•	•	•	•	•
	SH 124(95)	•			33445(60)		
	GHC4307(75)		21370(70)		•		
	PH 125C(70)		•				
	WEH5403(65)		09711(60)				
	DH A310(60)		•				
	GHC4215(53)						
	PH A18A(50)	73150(50)	76321(50)	80383(50)		85219(50)	85370(50)
	PH A18B(50)	•	•	•		•	•
	PH A18C(50)			48240(60)			
	SH 214(45)			•			
	SH 219(45)			•			
	WEH5302(40)						
	WEH6423(35)						
	WEH8427(35)						
	BH 235A(35)						
	BH 235B(35)						
	BH 237B(35)						
	BH 255A(35)						
	DH 1211(35)						

	DH 1217(35)						
	WEH5320(30)						
	DH 1117(30)						
	GHC4301(28)						
	WEH5304(20)						
3rd	UC Mc.(445)	**15212(150)**	85102(200)	06323(180)	21127(200)	06423(180)	06221(180)
	DH 2210(278)	18220(90)	09217(90)	21111(100)	12335(90)	**21256(100)**	76270(100)
	DH 2315(258)	09106(90)	06422(90)			12355(90)	82171(90)
	PH 100(234)	18491(90)	18450(90)	18342(90)	18320(90)		
	HH 1000(126)	48550(60)	51271(60)	21257(50)	70100(50)	70207(50)	70371(50)
	MM 103(115)	73359(50)	73270(50)	67250(50)	67306(50)	67371(50)	70414(50)
	MM A14(110)		36350(50)	73253(50)	70492(50)	70440(50)	
	DH 1212(107)	73310(50)			36225(50)		33124(50)
	DH 1112(99)						
	HH B103(96)			12212(90)	06321(90)	18290(90)	18240(90)
	HH B131(96)			•	•	•	•
	SH 124(95)						
	GHC4307(75)						
	PH 125C(70)						
	WEH5403(65)						
	DH A310(60)						
	GHC4215(53)						
	PH A18A(50)	36463(50)	73497(50)				70391(50)
	PH A18B(50)	•	•	36309(50)		36220(50)	•
	PH A18C(50)			•		•	
	SH 214(45)						
	SH 219(45)						
	WEH5302(40)						
	WEH6423(35)						
	WEH8427(35)						
	BH 235A(35)						
	BH 235B(35)						
	BH 237B(35)						
	BH 255A(35)						
	DH 1211(35)						

	DH 1217(35)						
	WEH5320(30)						
	DH 1117(30)						
	GHC4301(28)						
	WEH5304(20)						

(b)

Per. /Day	Room (Capacity)	1	2	3	4	5	6
1st	UC Mc.(445)	15110(800)	**03121(480)**	21120(830)	21259(460)	79104(600)	09105(560)
	D2210(278)	•	•	•	•	•	•
	DH2315(258)	•	•	•	•	•	•
	PH 100(234)	•	09344(60)	•	42401(180)	•	•
	HH1000(126)	•	03201(60)	•	•	15381(150)	**80226(180)**
	MM 103(115)	•		•	42101(180)	•	•
	MMA14(110)	•		•	•	•	•
	DH1212(107)	15451(150)		•	•	03325(60)	
	DH 1112(99)	•		15410(150)	03301(60)	•	
	HH B103(96)	•	03330(60)	•	•	09219(60)	03401(60)
	HH B131(96)	21600(70)	•	•	09221(60)	•	•
	SH 124(95)	•		03231(60)	•		
	GHC4307(75)	09507(60)		•	57480(30)		
	PH 125C(70)	•					
	WEH5403(65)	57152(30)					
	DH A310(60)		57173(30)				82271(30)
	GHC4215(53)						
	PH A18A(50)			09231(60)		88377(50)	88412(50)
	PH A18B(50)			•		•	•
	PH A18C(50)			•			
	SH 214(45)						
	SH 219(45)						
	WEH5302(40)	15110					
	WEH6423(35)						
	WEH8427(35)						
	BH 235A(35)						

	BH 235B(35)						
	BH 237B(35)						
	BH 255A(35)						
	DH 1211(35)						
	DH 1217(35)						
	WEH5320(30)						
	DH 1117(30)			15410	42101		
	GHC4301(28)						
	WEH5304(20)						80226
2nd	UC Mc.(445)	70122(200)	21122(460)	36225(220)	36201(400)	82141(120)	85102(110)
	DH 2210(278)	**15123(150)**	•	21228(130)	•	**21241(250)**	33106(270)
	DH 2315(258)	•	•	24351(90)	•	•	•
	PH 100(234)	24302(90)	73347(100)	33211(60)	24424(90)	73100(100)	36220(90)
	HH 1000(126)	21476(70)	24322(90)	33331(60)	33231(60)	24451(90)	•
	MM 103(115)	21369(70)	•	88302(50)	85211(50)	•	42444(90)
	MM A14(110)	•	33111(300)		88122(50)	85421(50)	•
	DH 1212(107)	09518(60)	•		21355(70)	21341(70)	85241(50)
	DH 1112(99)	•	•		•	•	
	HH B103(96)	21476		48210(60)	33301(60)	33341(60)	33338(60)
	HH B131(96)	48315(60)	•	•	•	•	•
	SH 124(95)	•			33445(60)		
	GHC4307(75)		21370(70)		•		
	PH 125C(70)		•				
	WEH5403(65)		09711(60)				
	DH A310(60)		•				
	GHC4215(53)						
	PH A18A(50)	73150(50)	76321(50)	80383(50)		85219(50)	85370(50)
	PH A18B(50)	•	•	•		•	•
	PH A18C(50)			48240(60)			
	SH 214(45)			•			
	SH 219(45)			•			
	WEH5302(40)						
	WEH6423(35)						
	WEH8427(35)						
	BH 235A(35)						

	BH 235B(35)						
	BH 237B(35)						
	BH 255A(35)						
	DH 1211(35)						
	DH 1217(35)						
	WEH5320(30)						
	DH 1117(30)						
	GHC4301(28)						
	WEH5304(20)						
3rd	UC Mc.(445)	**21256(100)**	85102(200)	06323(180)	21127(200)	06423(180)	06221(180)
	DH 2210(278)	18220(90)	09217(90)	21111(100)	12335(90)		76270(100)
	DH 2315(258)	09106(90)	06422(90)			12355(90)	82171(90)
	PH 100(234)	18491(90)	18450(90)	18342(90)	18320(90)		
	HH 1000(126)	48550(60)	51271(60)	21257(50)	70100(50)	70207(50)	70371(50)
	MM 103(115)	73359(50)	73270(50)	67250(50)	67306(50)	67371(50)	70414(50)
	MM A14(110)		36350(50)	73253(50)	70492(50)	70440(50)	
	DH 1212(107)	73310(50)		**15212(150)**	36225(50)		33124(50)
	DH 1112(99)			•			
	HH B103(96)			12212(90)	06321(90)	18290(90)	18240(90)
	HH B131(96)			•	•	•	•
	SH 124(95)						
	GHC4307(75)						
	PH 125C(70)						
	WEH5403(65)						
	DH A310(60)			15212			
	GHC4215(53)						
	PH A18A(50)	36463(50)	73497(50)				70391(50)
	PH A18B(50)	•	•	36309(50)		36220(50)	•
	PH A18C(50)			•		•	
	SH 214(45)						
	SH 219(45)						
	WEH5302(40)						
	WEH6423(35)						
	WEH8427(35)						
	BH 235A(35)						

	BH 235B(35)						
	BH 237B(35)						
	BH 255A(35)						
	DH 1211(35)						
	DH 1217(35)		15212				
	WEH5320(30)						
	DH 1117(30)						
	GHC4301(28)						
	WEH5304(20)						