CourseComb Design Document

Team

- HJ Suh: hjsuh@princeton.edu
- Ada Nguyen: lbn@princeton.edu
- Matt Yi: mkyi@princeton.edu
- DoWon Kim dowon@princeton.edu

1. Overview
Students spend countless hours trying to perfect their course schedule to their liking. There are simply too many different combinations of courses and class times (including precepts, labs, etc.) to consider and it is almost impossible for students to pick their optimal schedule. We plan to create a new quick and simple way to select courses. CourseComb is an intuitive solution to a problem that every student faces at the start of every semester.

We have modeled CourseComb to emulate the natural way in which students approach course selection. The user inputs a long list of courses of interest and selects various filtering options (e.g. number of courses, must-take courses, no Friday classes) to narrow down the search. The website gives a list of all possible combinations of courses. If the user clicks on a certain combination, the website shows a list of all possible combinations of different class times. The user can click on their favorite one and voila! - they end up with the perfect schedule in just 3 simple steps!

2. Requirements and Target Audiences
Our primary target audience is Princeton students who are trying to work out their course schedule. The website will require CAS authentication. Although it is ideal to allow prefrosh and others interested to use our website, we will restrict access to only CAS-authenticated Princeton students and staff because we plan to include information private to the Princeton community, such as course evaluations. We hope that CourseComb will be the go-to website for all students when selecting courses, regardless of whether they are a senior or freshman, and whether they are someone who likes to shop 10+ courses at the start of semester or someone who knows exactly what they're taking years ahead of time.

Existing Solutions:
Tigerhub: Tigerhub has many advantages over student-created websites since it is university-run and therefore acts quite literally as a "hub" for anything academically-related on campus. The course selection feature of the website allows users to add, drop, and change courses on an interactive calendar in a relatively easy way. The best feature of Tigerhub is that

once the user creates the final schedule, they can directly upload the courses to the Course Queue, which allows them to directly enroll in these courses at a designated time.

<u>ReCal:</u> ReCal is probably the most commonly used course scheduling website on campus. It offers a cleaner and faster way to select courses than TigerHub's course scheduler. Besides the ease of use, one important feature of ReCal is the ability to export the schedule to Google Calendar.

We see multiple problems with the two options listed above. First, the two existing solutions are good ways to visualize the times of all classes offered, but users often have a hard time filtering through the huge number of choices they are faced with on a cluttered weekly calendar (Figure 1). The two websites offer no powerful tools with which users can easily narrow down their options to just one schedule. It is often the case that users go through endless cycles of adding, deleting, and changing courses and class times. As good as ReCal is, this can be rather stressful and unproductive. Furthermore, the developers of ReCal have left campus, which means maintenance of the website is not guaranteed.

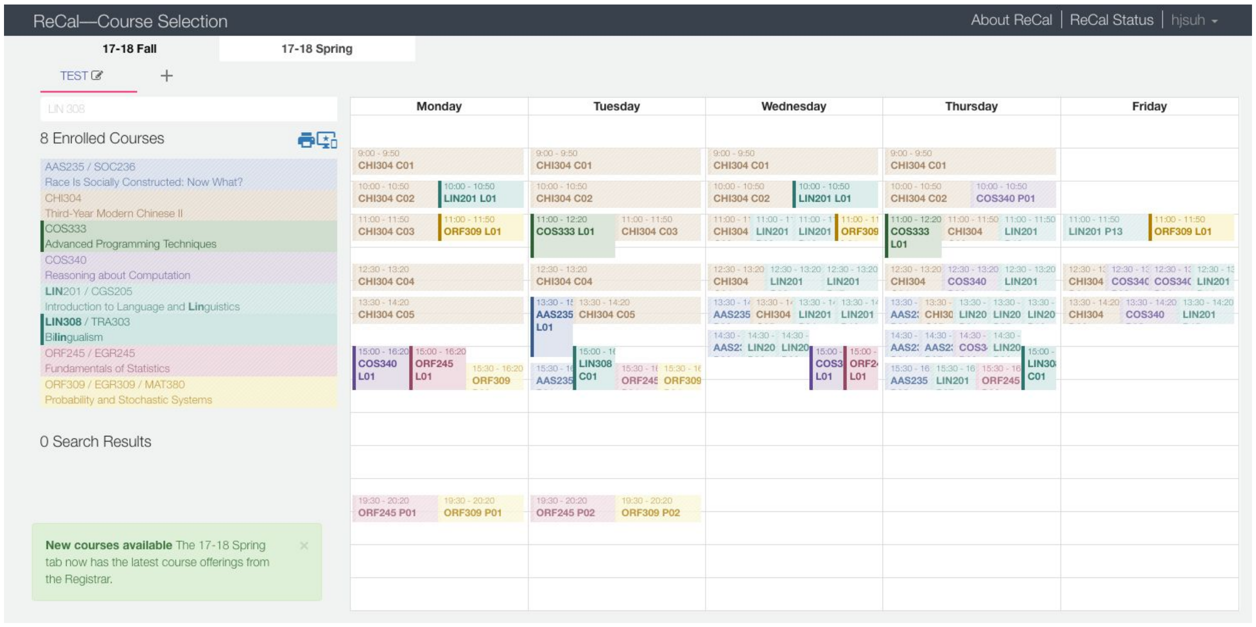

ReCal—Course Selection
About ReCal
ReCal Status
hjsuh
17-18 Fall
17-18 Spring
TEST
8 Enrolled Courses
AAS235 / SOC236
Race Is Socially Constructed: Now What?
CHI304
Third-Year Modern Chinese II
COS333
Advanced Programming Techniques
COS340
Reasoning about Computation
LIN201 / CGS205
Introduction to Language and Linguistics
LIN308 / TRA303
Bilingualism
ORF245 / EGR245
Fundamentals of Statistics
ORF309 / EGR309 / MAT380
Probability and Stochastic Systems
0 Search Results
New courses available The 17-18 Spring tab now has the latest course offerings from the Registrar.
Monday
Tuesday
Wednesday
Thursday
Friday

Figure 1. Cluttered ReCal Calendar

Our plan is to incorporate the best parts of both TigerHub and ReCal into our website while improving on the negatives. We separate the process of "combing" through course options into two separate steps of choosing courses and choosing class times. Users can specify their preferences in courses or class times through these options:

- Must-have courses e.g. COS 333, SOC 101
- Must-have distributions e.g. LA, HA, STN
- Must-have departments e.g. COS, MAT
- Class time preferences e.g. No Friday class, No evening class, After 10am

- Exclude fully-enrolled classes
- Include 1 PDF-only class

Our website will intelligently give a ranked list of all possible combinations of courses and class times taking into account the user's preferences. At the end, the user should be able to favorite and save their preferred schedules.

Stretch Goals

- Order in which courses are entered represent weighting/importance of course and the search results are appropriately ranked
- Allow users upload their final course schedule to the TigerHub Course Queue and Google Calendar
- As a filter option, have when-is-good-style time specification of preferred class times for more flexible customization
- Shopping period mode, which is outlined in more detail in use case 3 in functionality
- Rank the course combinations by the average rating of the courses

3. Functionality

Our basic structure is not restrictive and gives the user freedom to make the best use of the tools provided as they see fit.

- **Use case 1:** John has 10 courses of interest and wants to narrow them down to 4 courses. He knows that he is going to take COS 333 and ORF 309, and so he inputs the 10 courses into CourseComb and enters the must-have courses on the website. He also needs to fulfil an HA requirement and sets the filters accordingly. When he presses "Create My Schedule," CourseComb shows 4 different combinations of courses that are possible to take together. From these 4 options, John picks his favorite and saves it to his list of favorite course combinations.
- **Use case 2:** John now wants to choose the optimal class times for his 4 courses. John updates the filters such that he doesn't get Friday classes or classes before 10am. The website dynamically changes to take this change of filters into account and when John clicks on a certain course combination, all possible class times for his chosen course combination are shown. John likes to have his classes evened out over the week, so he deletes some options where his Tuesdays and Thursdays have 6 classes. He also wants time for lunch, and so deletes schedules with 12:30 class. After comparing the different schedules, John picks his favorite and saves it to his list of favorite course schedules.
- **Use case 3 (stretch goal):** Alice is a freshman who is so excited to take her first classes at Princeton that she has a list of 15 courses that she wants to take. Her academic advisor tells her that it is best to limit the number of courses to 4 or 5, and so Alice decides to attend every one of her 15 courses in shopping period to help her choose. Alice enters the 15 courses into CourseComb, checks the 'Shopping Period Mode' box and presses 'Create My Schedule'. CourseComb shows Alice the schedule she needs to follow to try out all of her courses. Alice exports this schedule to Google Calendar and embarks on her first (busy) week at Princeton.

4. Design

User Interface:
The sketches in the Appendix show a good representation of the structure of our website. The below description acts as a clarification to specify how the user interface changes as the user interacts with it.

We plan to implement the user interface with React. All the information will be on one page with three sections:

- The leftmost sidebar will allow users to input desired courses and the number of courses they plan on taking. Since we plan to implement multiple filters, instead of cluttering the sidebar with options, we have a pop-up page that comes up when the user clicks on the filter icon.
- The middle bar will display all different possible combinations of courses listed. Users can dynamically delete unwanted combinations as they see fit. In addition, users can favorite certain course combinations for later reference.
- The rightmost bar will show all different possible combinations of classes (including precept, lab, etc.) when a certain course combination is clicked. Just as with course combinations, users can delete unwanted options and favorite to save certain schedules. When the user clicks on the '+' button on one of the options, an image of what the timetable would look like slides down from the option so that the user has a visualization.

Process:
When the user inputs courses of interest and enters filters for the course search, an API call is made with a dictionary of all information inputted. We construct a QuerySet and go through the dictionary to filter the data we query from the database (e.g. 'No Friday Class' filter checked means database query does not return any courses with Friday class).

From the selected data, we generate all possible combinations of courses, taking the filters and times of classes into account. We will detect time conflicts between courses only by considering the times of Lecture (L), Class (C), Session (S/U), since we are doing a first 'screening' to see if the courses are compatible. Taking all precept/lab times into account would require a lot of computing power, and usually multiple precept/lab times are offered that they are unlikely to be restrictive.

When the user clicks on a certain course combination, we go through a similar process as outlined above but with the focus on displaying different combinations of class times, including lab (B), drill (D), ear training (E), film (F), and precept (P), rather than different combinations of courses.

We pass the processed data in JSON via RESTful API to the frontend. Ideally, we will make the search results dynamically change as filters are changed after the first search is made, i.e. the second and third bars of the website would dynamically change as the user clicks on "Save Changes" on the filter. The frontend will need to use AJAX calls to update the page.

Scraping:
Luckily, there have been numerous projects related to Princeton courses, so we are unlikely to have to write a lot of code to create our own scraper. scraper.py from assignment 4, ReCal and ReCourse's web scrapers will be used as references to obtain the data needed. In order to keep an accurate account of class enrollment, we will periodically scrape Registrar website to update the data. The fields we need are: course title, course id, department(s) and number(s), distribution area, prerequisites, description, classes (class number, people enrolled, enrollment limit, start and end time, days, section, building, room number), PDF-ability of class, course rating, a link to course information page on registrar, a link to the course's evaluation page.

Data Management:
We will use PostgreSQL for our database. Besides the course data obtained through scraping, we also will need to store user's information, including their favorited schedules and deleted combinations. The database and the website will both be hosted on Heroku.

Stretch Goal: when a user returns to the website and wants to continue working from the selections they made previously, the website should give an option to upload previous 'works in progress'. This would mean that the database needs to store saved 'works in progress' for each user.

5. Timeline

March 19 - March 25 (Spring break):

- Learn technologies: React.js, Django, PostgreSQL, Heroku, CAS Authentication
- Send out a simple survey to gauge student interest in product
- Set up GitHub and become familiar with basic operations
- Finish Assignment 5, so that focus can be on project when we return

March 26 - April 1

- Attend Hackathon together to work on project (March 30 - April 1)
- Edit scraper code to gather all data that we need for our website
- Set up basic UI: allows users to enter course name and the query is passed to the backend; display the information that backend returns in JSON
- Put Spring 2018 courses into database and make basic queries about course information; receive query from user and returns course information in JSON

April 2 - April 8

- Complete scraper and link with database
- Develop the logic for finding all combinations of listed courses and class times
- Develop the user interface such that all components of the website can be displayed

April 9 - April 15

- Implement core filters
- Implement scraper such that it scrapes periodically to update changes, especially to keep class enrollment up to date
- Implement display of calendar when a certain 'precept combination' is clicked
- Develop the logic for finding timing conflicts between different courses and class times

April 13 : Project Prototype
- Have core features of website running (not necessary to have all filters implemented)

April 16 - April 22
- Finish implementing all filters
- Implement favoriting course combinations, final schedules, saving works in progress
- Final fine tuning of UI

April 23 - April 29
- Work on stretch goals: course importance weighting, when-is-good style time specification, shopping period mode, ranking by the average rating of courses selected
- Thorough testing of alpha test product and debugging

April 27 : Alpha Test
- We should have something close to a final product other than minor bugs

April 30 - May 6
- Exportation to Google Calendar and TigerHub Course Queue
- Launch product to Princeton students to receive feedback

May 4 : Beta Test
- Our product should be fully operational without any bugs

May 7 - May 12
- If necessary, some last minute changes and debugging after undergoing thorough testing
- Take feedback into account and make necessary changes

May 9, 10 : Demo Days
May 13 : Project Deadline

6. Risks and Outcomes

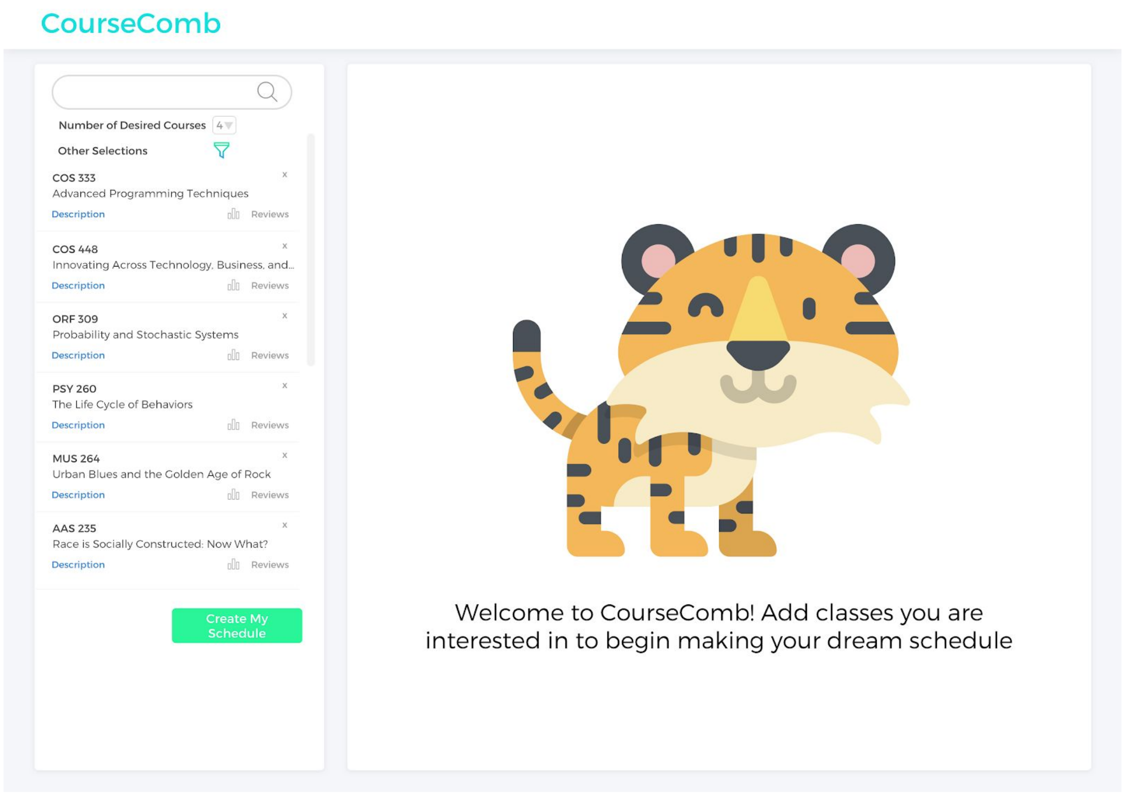
The biggest risk is that the number of combinations will be overwhelmingly large, leading to either extremely slow speed or a cluttered UI. We want to provide users with powerful tools to narrow down their search from hundreds of combinations in intuitive and efficient ways. The biggest challenge will be to make these tools simple, yet powerful.

Another challenge is to make the user interface as intuitive as possible. After playing around for a couple of minutes, the user should know exactly what each part of the website does. A possible solution could be to include a ‘help’ page, or have labels for each part of our website. We are lucky to have willing users on campus, and so receiving feedback from real users will be most helpful in improving our product.

We are using React for our frontend, and are worried that the learning curve will be too steep. As beginners in web development, we might switch to easier options such as Bootstrap or JQuery.

Lastly, we are all students who take mostly computational courses, so it is important for us seek the perspective of humanities students who may have different preferences when it comes to course selection.

7. Appendix



CourseComb

Number of Desired Courses 4

Other Selections

COS 333
Advanced Programming Techniques
Description · Reviews

COS 448
Innovating Across Technology, Business, and...
Description · Reviews

ORF 309
Probability and Stochastic Systems
Description · Reviews

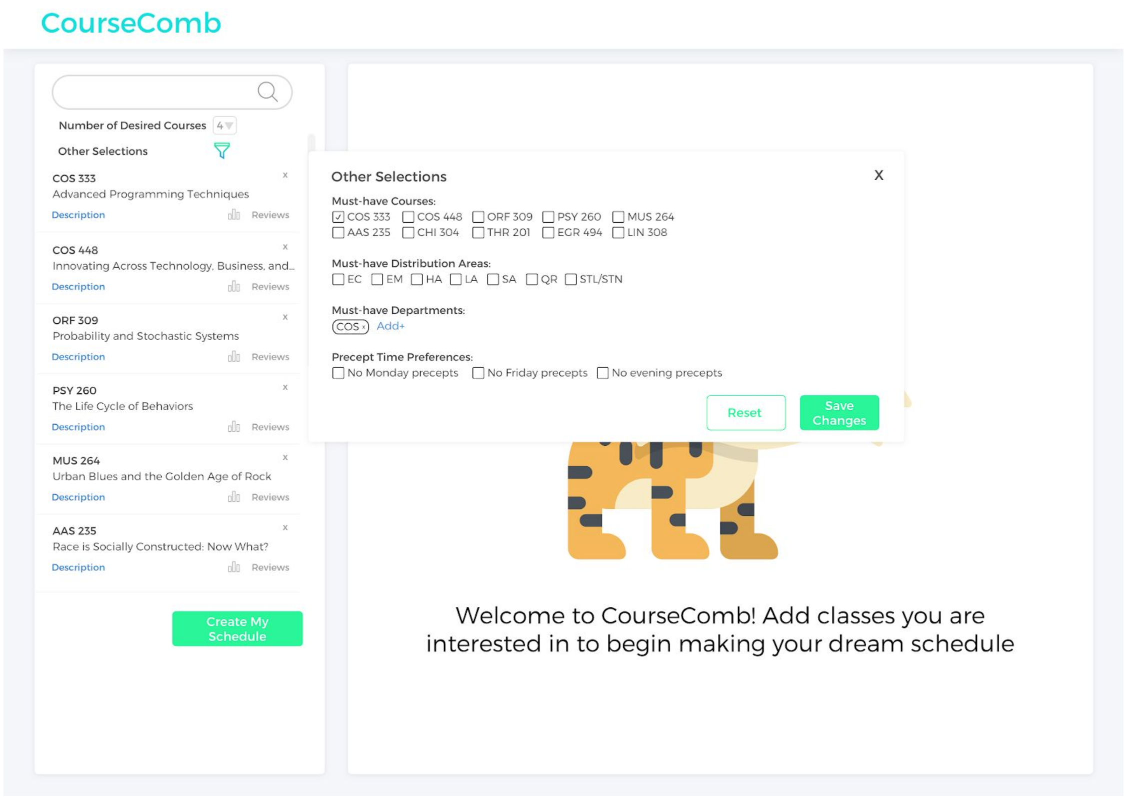
PSY 260
The Life Cycle of Behaviors
Description · Reviews

MUS 264
Urban Blues and the Golden Age of Rock
Description · Reviews

AAS 235
Race is Socially Constructed: Now What?
Description · Reviews

Create My Schedule

Welcome to CourseComb! Add classes you are interested in to begin making your dream schedule



CourseComb

Number of Desired Courses 4

Other Selections

COS 333
Advanced Programming Techniques
Description · Reviews

COS 448
Innovating Across Technology, Business, and...
Description · Reviews

ORF 309
Probability and Stochastic Systems
Description · Reviews

PSY 260
The Life Cycle of Behaviors
Description · Reviews

MUS 264
Urban Blues and the Golden Age of Rock
Description · Reviews

AAS 235
Race is Socially Constructed: Now What?
Description · Reviews

Create My Schedule

Other Selections X

Must-have Courses:
☑ COS 333 ☐ COS 448 ☐ ORF 309 ☐ PSY 260 ☐ MUS 264
☐ AAS 235 ☐ CHI 304 ☐ THR 201 ☐ EGR 494 ☐ LIN 308

Must-have Distribution Areas:
☐ EC ☐ EM ☐ HA ☐ LA ☐ SA ☐ QR ☐ STL/STN

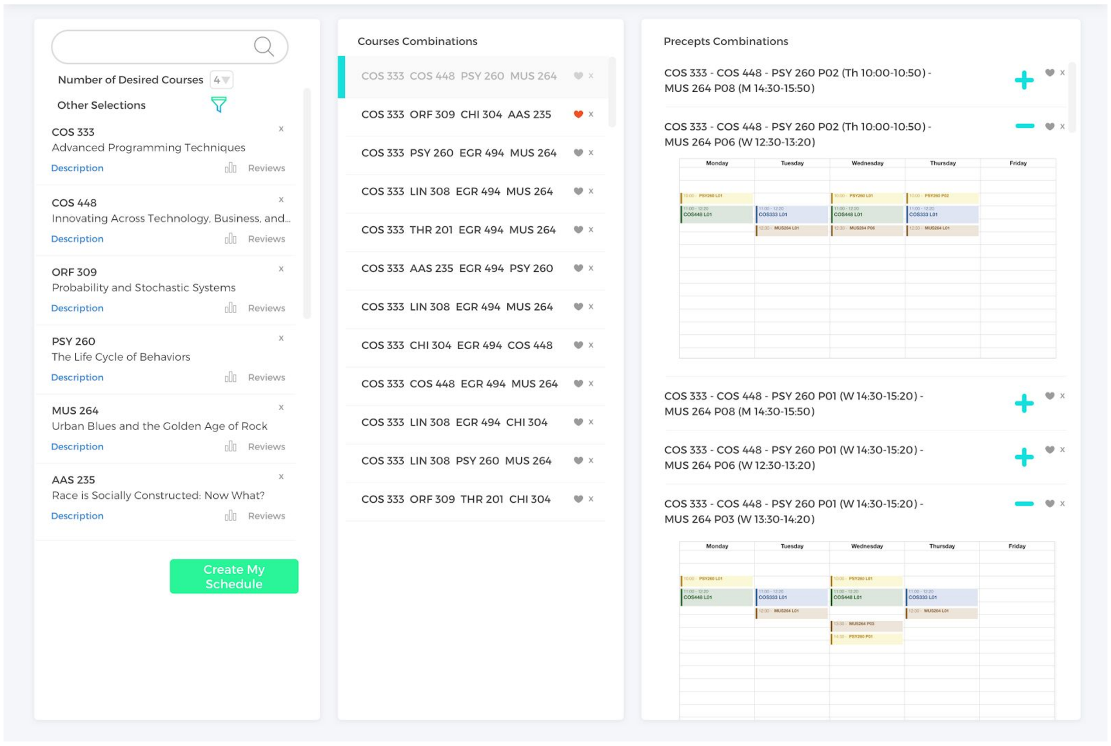
Must-have Departments:
COS Add+

Precept Time Preferences:
☐ No Monday precepts ☐ No Friday precepts ☐ No evening precepts

Reset · Save Changes

Welcome to CourseComb! Add classes you are interested in to begin making your dream schedule

CourseComb



Number of Desired Courses 4

Other Selections

COS 333
Advanced Programming Techniques
Description Reviews

COS 448
Innovating Across Technology, Business, and...
Description Reviews

ORF 309
Probability and Stochastic Systems
Description Reviews

PSY 260
The Life Cycle of Behaviors
Description Reviews

MUS 264
Urban Blues and the Golden Age of Rock
Description Reviews

AAS 235
Race is Socially Constructed: Now What?
Description Reviews

Create My Schedule

Courses Combinations

COS 333 COS 448 PSY 260 MUS 264
COS 333 ORF 309 CHI 304 AAS 235
COS 333 PSY 260 EGR 494 MUS 264
COS 333 LIN 308 EGR 494 MUS 264
COS 333 THR 201 EGR 494 MUS 264
COS 333 AAS 235 EGR 494 PSY 260
COS 333 LIN 308 EGR 494 MUS 264
COS 333 CHI 304 EGR 494 COS 448
COS 333 COS 448 EGR 494 MUS 264
COS 333 LIN 308 EGR 494 CHI 304
COS 333 LIN 308 PSY 260 MUS 264
COS 333 ORF 309 THR 201 CHI 304

Precepts Combinations

COS 333 - COS 448 - PSY 260 P02 (Th 10:00-10:50) - MUS 264 P08 (M 14:30-15:50)

COS 333 - COS 448 - PSY 260 P02 (Th 10:00-10:50) - MUS 264 P06 (W 12:30-13:20)

Monday	Tuesday	Wednesday	Thursday	Friday
10:00 - PSY260 L01		10:00 - PSY260 L01	10:00 - PSY260 P02	
11:00 - 12:20 COS448 L01	11:00 - 12:20 COS333 L01	11:00 - 12:20 COS448 L01	11:00 - 12:20 COS333 L01	
	12:30 - MUS264 L01	12:30 - MUS264 P06	12:30 - MUS264 L01	

COS 333 - COS 448 - PSY 260 P01 (W 14:30-15:20) - MUS 264 P08 (M 14:30-15:50)

COS 333 - COS 448 - PSY 260 P01 (W 14:30-15:20) - MUS 264 P06 (W 12:30-13:20)

COS 333 - COS 448 - PSY 260 P01 (W 14:30-15:20) - MUS 264 P03 (W 13:30-14:20)

Monday	Tuesday	Wednesday	Thursday	Friday
10:00 - PSY260 L01		10:00 - PSY260 L01		
11:00 - 12:20 COS448 L01	11:00 - 12:20 COS333 L01	11:00 - 12:20 COS448 L01	11:00 - 12:20 COS333 L01	
	12:30 - MUS264 L01		12:30 - MUS264 L01	
		13:30 - MUS264 P03		
		14:30 - PSY260 P01		