CS-639 Building User Interfaces, Fall 2020, Professor Mutlu

React 2 β (3 Points)

Improving Usability Using Heuristic Evaluation

In this assignment, you will put the ten usability heuristics we learned in class into practice toward improving the usability of your *React 2 α* deliverable. You will focus on specific components of your design, identify potential violations of the heuristics, make design recommendations to address these violations, and implement recommendations that are feasible to create a new deliverable. Use this opportunity to make concrete design decisions about your project, to improve your design using the heuristics, and to build a keen eye for identifying usability issues as a UX developer.

Step 1—Identify A Focus. (0.2 Points) Review your *React 2 α* deliverable with a critical eye to identify 3–5 "components" that you think are most consequential for user experience.

Step 2—Review the Heuristics. Review the ten usability heuristics we discussed in class from the slides, what principle each heuristic represents, and examples of the violations of the heuristics.

Step 3—Identify Potential Violations. (1.0 Points) Focusing on your components, inspect your design, considering each usability heuristic, for any violations of the heuristics.

Step 4—Develop Design Recommendations. (0.4 Points) For each violation you identified in the previous step, provide a design recommendation for addressing it, assessing its feasibility.

1. Visibility of system status: display the number of search results in the sidebar after the user sets the filter
2. Match between system and real world: start the numbering with 1, and change the time into natural language with the first letter of the week days capitalized

Step 5—Implement Your Recommendations. (1.4 Points) Implement the design recommendations that you identified as "feasible" in the previous step in your prototype, updating your design.

Submission Details

GitHub Classroom Starter Code

React 2 β will build on your implementation of React 2 α. You should copy your code from your React 2 α project to the React 2 β repository linked above, as that will be your starter code. When you commit and push, ensure that you are committing and pushing to the react2-beta repository, not react2-alpha.

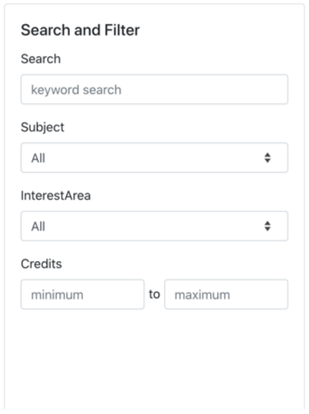
To complete the assignment, you will need to submit a completed version of this document as PDF to Canvas. In addition, you will submit your repository name and latest commit hash from GitHub Classroom, e.g. react2-beta-ctnelson1997, 2b0ef83.

Step 1. Identify A Focus. (0.2 Points)

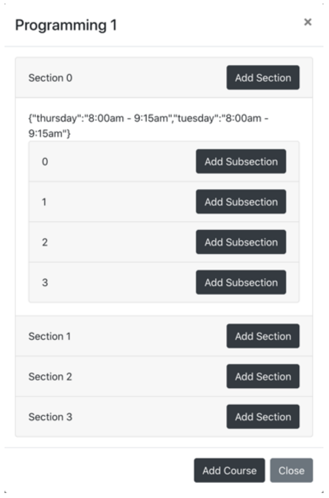
In this step, you will review your *React 2 α* deliverable with a critical eye to identify 3–5 "components" that you think are most consequential for user experience and that you will put under the microscope of heuristic evaluation in the next step. In real life, your application might have hundreds of components, screens, or pages, and you will have to focus your efforts on a limited set that will make the most difference in terms of effectiveness and user experience. Similarly, you will review your design and identify 3–5 components to focus on. Here, a "component" can be the entire page/view (e.g., recommended courses) or a reusable component (e.g., the course component, the rating component), but not something as small as a button or label. Provide screenshots of each component below and provide a brief justification (1–2 sentences) of why you think each one is a critical component.

---

1. Search bar – enables user to search for courses with specifications, effectively narrow down the scope of target courses.

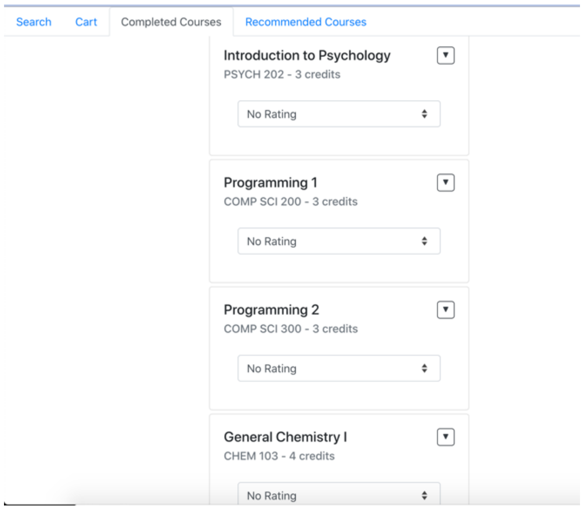


2. Course component – displays necessary information about a course and enables user to decide whether or not add a course or any of its sections/subsections to cart and whether the requisites for taking a specific course are met. It's critical because it provides the pivotal functionality of this application.



{"thursday":"8:00am - 9:15am","tuesday":"8:00am - 9:15am"}

3. Completed courses – enables user to rate courses already been taken, which provides necessary information for the recommender algorithm



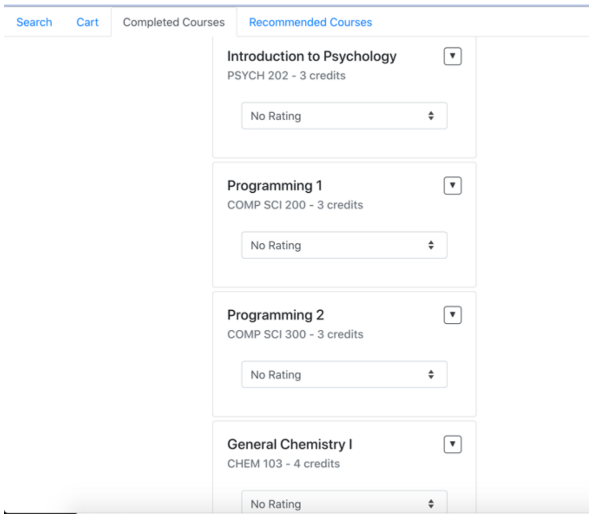
Introduction to Psychology
PSYCH 202 - 3 credits
No Rating

Programming 1
COMP SCI 200 - 3 credits
No Rating

Programming 2
COMP SCI 300 - 3 credits
No Rating

General Chemistry I
CHEM 103 - 4 credits
No Rating

4. Recommended courses – target of the recommender algorithm, adds individualized recommendations for the user given the courses taken



Introduction to Psychology
PSYCH 202 - 3 credits
No Rating

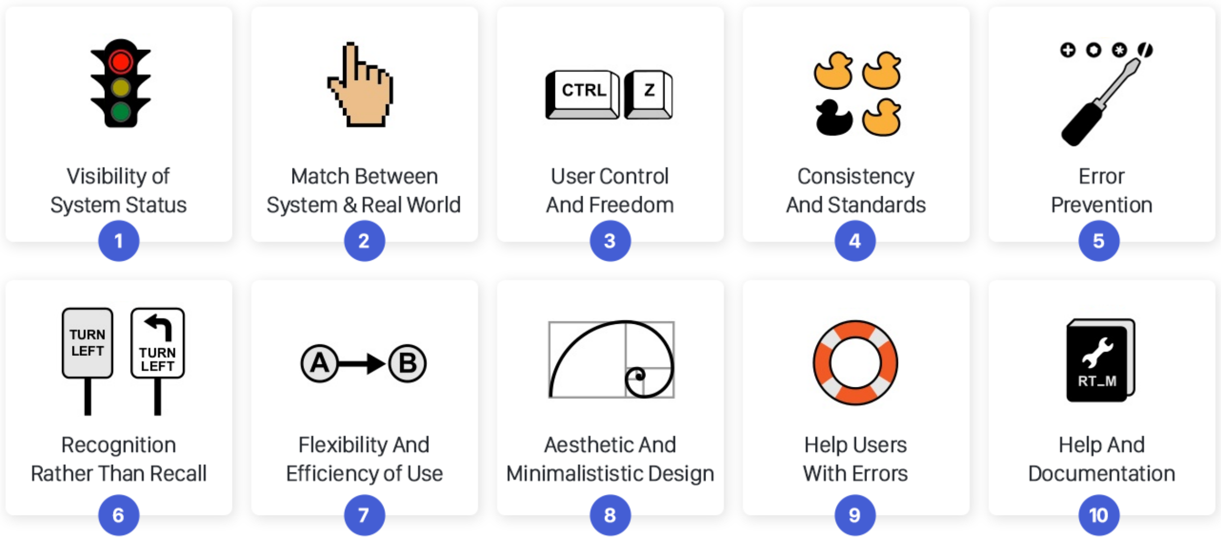
Programming 1
COMP SCI 200 - 3 credits
No Rating

Programming 2
COMP SCI 300 - 3 credits
No Rating

General Chemistry I
CHEM 103 - 4 credits
No Rating

Step 2. Review the Heuristics.

Carefully review the ten usability heuristics we discussed in class from the slides, what principle each heuristic represents, and examples of the designs that violate and support the heuristics. Below is a cheat sheet for Nielsen's ten heuristics that you can use in the next step. (This step does not have any deliverables.)



Visibility of System Status 1
Match Between System & Real World 2
CTRL Z
User Control And Freedom 3
Consistency And Standards 4
Error Prevention 5
TURN LEFT
TURN LEFT
Recognition Rather Than Recall 6
A B
Flexibility And Efficiency of Use 7
Aesthetic And Minimalistic Design 8
Help Users With Errors 9
RT_M
Help And Documentation 10

Image source: UX Collective

Step 3. Identify Potential Violations. (1.0 Points)

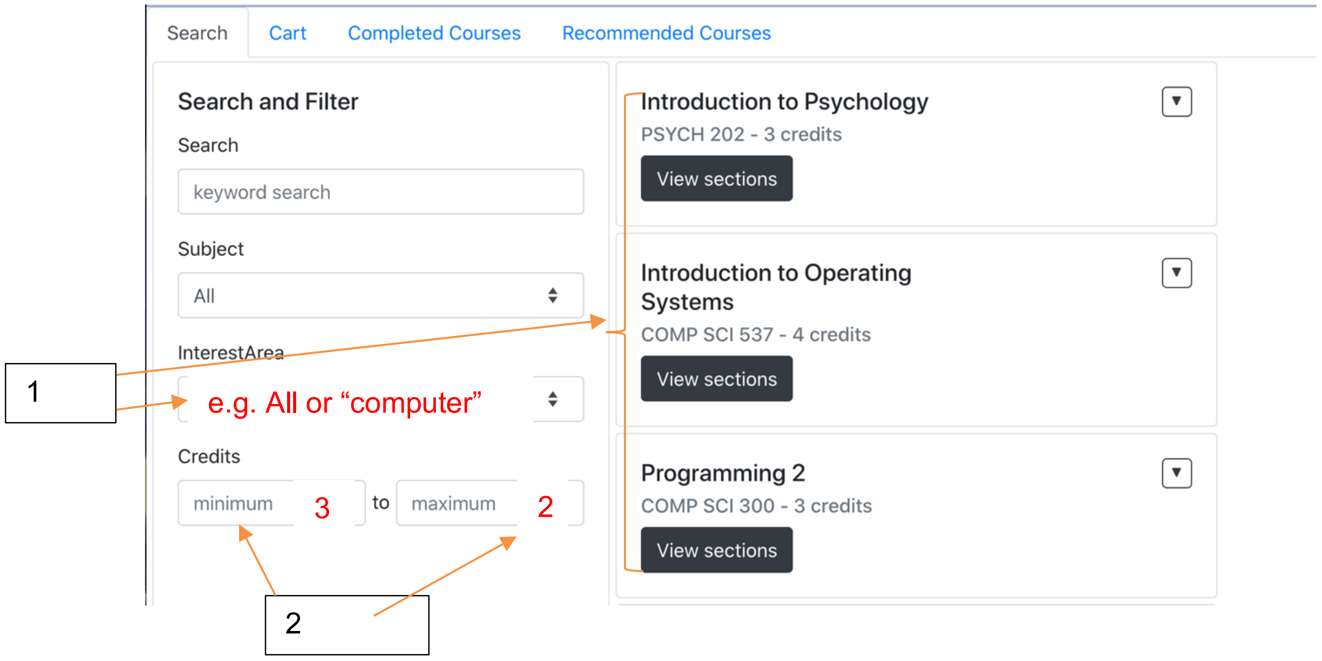
Focusing on your components, inspect your design, considering each usability heuristic, for any violations of the heuristics. For each violation, use the following table to briefly describe the violation and give it a unique number (specified in the # column). Make copies of your screenshots from Step 1, focusing on the design elements you are considering in this step, and mark them with the unique numbers so that the reader of your report can find the location of the violation in the screenshots and read your description in the table below. In addition, color-code the violations for severity, highlighting with red, orange, yellow, green, and gray for the severity-rating scale we covered in class (with red being most severe to gray being a non-issue).

Heuristic	#	Component 1	#	Component 2	#	Component 3
Visibility of system status		1. User don't know if the search result is filtered if the first several courses stay the same after the search		No usability problem		5. User don't get informed about what happens after rating a completed course (recommendations might be updated)
Match between real world & system		No usability problem		3. The time is displayed in JSON format instead of the user's language. 4. Numbering of sections follows the computer science rule		No usability problem

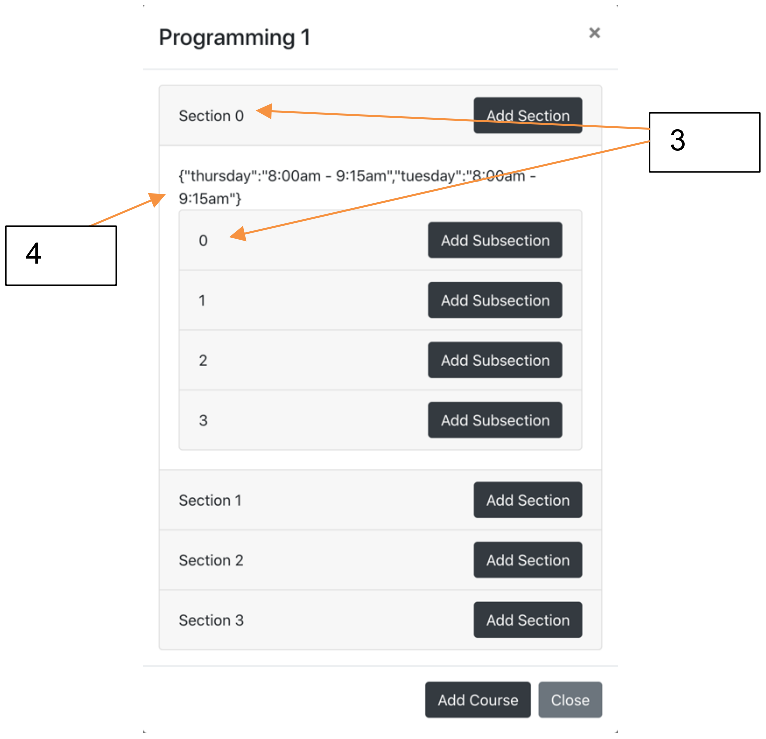
				of starting with 0 but a natural way if to start with 1.		
User control & freedom		No usability problem		No usability problem		No usability problem
Consistency & standards		No usability problem		No usability problem		No usability problem
Error prevention		2. The maximum credits can't be more than the minimum, or there won't be any corresponding results		No usability problem		No usability problem
Recognition rather than recall		No usability problem		No usability problem		No usability problem
Flexibility & efficiency of use		No usability problem		No usability problem		No usability problem
Aesthetic & minimalist design		No usability problem		No usability problem		No usability problem
Help users with errors		2. The maximum credits can't be more than the minimum, or there won't be any corresponding results		No usability problem		No usability problem
Help & documentation		No usability problem		6. Documentation could be given to explain the process of adding courses – e.g. adding a course means adding all its sections and subsections		7. Documentation could be given to explain that the drop-down is to rate their completed courses and this can affect the recommended courses for them

Heuristic	**#**	**Component 4**	**#**	**Component 5**	**#**	**Component 6**
Visibility of system status		No usability problem				
Match between real world & system		No usability problem				
User control & freedom		No usability problem				
Consistency & standards		No usability problem				
Error prevention		No usability problem				
Recognition rather than recall		No usability problem				
Flexibility & efficiency of use		No usability problem				
Aesthetic & minimalist design		No usability problem				
Help users with errors		No usability problem				
Help & documentation		No usability problem				

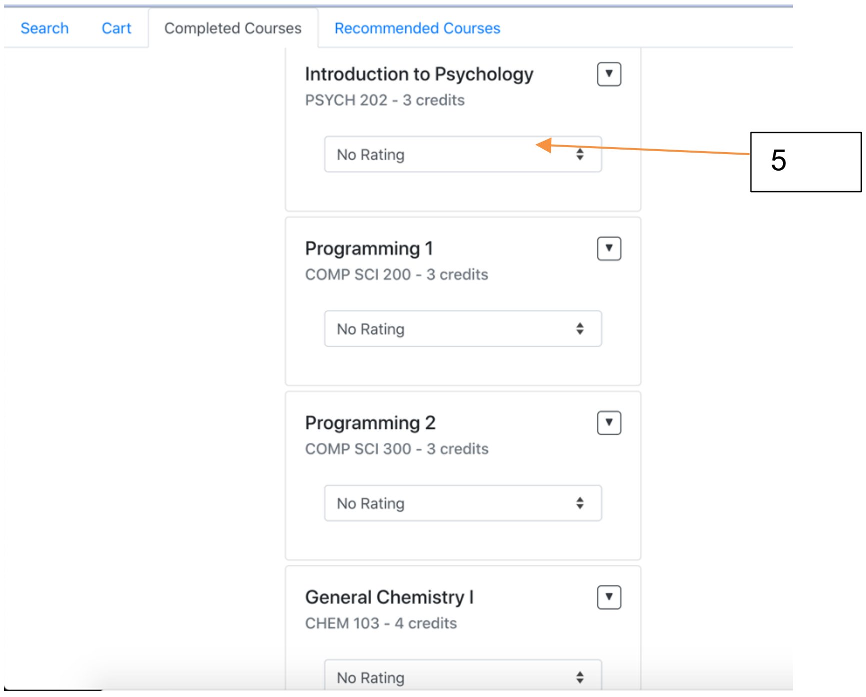
Component #1:



Component #2:



Component #3:



Introduction to Psychology
PSYCH 202 - 3 credits
No Rating

Programming 1
COMP SCI 200 - 3 credits
No Rating

Programming 2
COMP SCI 300 - 3 credits
No Rating

General Chemistry I
CHEM 103 - 4 credits
No Rating

Step 4. Develop Design Recommendations. (0.4 Points)

For each violation you identified in the previous step, provide a design recommendation for addressing it along with an indication of whether or not it is feasible to implement the recommendation as an extension of your *React 2 α* deliverable. (Only recommendations that are beyond the capabilities we learned in class or beyond the scope of the project should be marked as not being feasible.) Order the table of recommendations based on the severity of the usability problem from most severe to least severe. Use the table below to describe your recommendations, adding additional rows as needed, and follow the same color-coding from the previous step for severity ratings.

#	**Recommendation**	**Feasibility (Yes/No)**
1	Display the number of search results in the sidebar after the user sets the filter	Yes
3	Start the numbering with 1	Yes
4	Change the time display to natural language with the first letter of the week days capitalized	Yes
2	Alerts the user about the error when a larger number than maximum is entered as minimum	Yes

5	Notify the user that some courses are added to the recommended courses when a high rating is given for a completed course	Yes
6	Add a help tab to explain the usage and logic of adding courses, sections, and subsections	Yes
7	Add a help tab to explain how the ratings of completed courses can affect the recommended courses	Yes

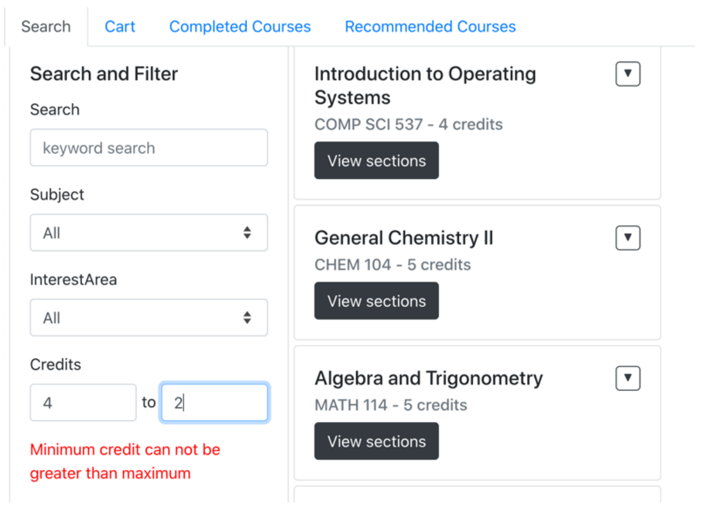
Step 5. Implement Your Recommendations. (1.4 Points)

In this step, you will implement the design recommendations that you identified as "feasible" in the previous step in your prototype, updating your design. To receive full points, you will implement at least three design recommendations that can improve one or more of the components you focused on. Submit your improved React project based on instructions below and provide a paragraph that summarizes the outcome of the heuristic evaluation. In this paragraph, reflect on how your design improved, what you learned about usability in the process of applying the heuristics, and whether you gained any unexpected insights about your design.

Your deliverable will be a completed version of this document, attached to the canvas assignment as a PDF, and the GitHub Classroom repository name and latest commit hash.

Refactoring the numbering matches the real-world convention of starting with 1. It makes the information of sections and subsections appear in a natural order instead of speaking the language only familiar to computer scientists (starting with 0). Refactoring the time display achieves the same goal of enabling the application to speak the user's language. Having an alert message to inform the user about the number of courses after the search filter is set makes it easier for user to get reasonable feedback and keep track of the status of the search. When the maximum credits entered is smaller than the minimum, the system is improved to help the user recognize that error and prompt recovering actions, instead of showing a blank page of results and letting the user to figure out the error. From this process, I find heuristic evaluation to be very useful in terms of making the application more user-friendly and figuring out small points that are easily missed during the initial development. Recognizing and addressing the small problems/violations contribute to the a more successful and user-oriented system.

See next page for screenshots of the implementation of resolving the violations.



Search | Cart | Completed Courses | Recommended Courses

Search and Filter

Search

keyword search

Subject

All

InterestArea

All

Credits

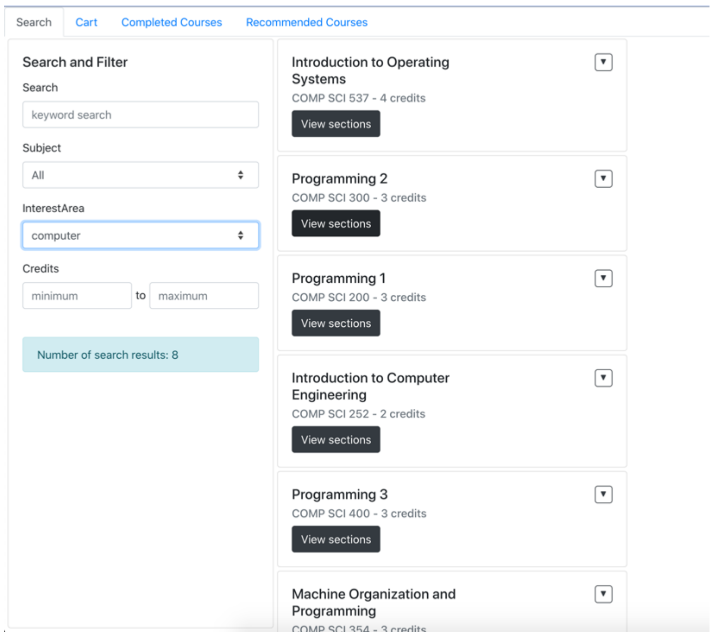
4 to 2

Minimum credit can not be greater than maximum

Introduction to Operating Systems
COMP SCI 537 - 4 credits
View sections

General Chemistry II
CHEM 104 - 5 credits
View sections

Algebra and Trigonometry
MATH 114 - 5 credits
View sections



Search | Cart | Completed Courses | Recommended Courses

Search and Filter

Search

keyword search

Subject

All

InterestArea

computer

Credits

minimum to maximum

Number of search results: 8

Introduction to Operating Systems
COMP SCI 537 - 4 credits
View sections

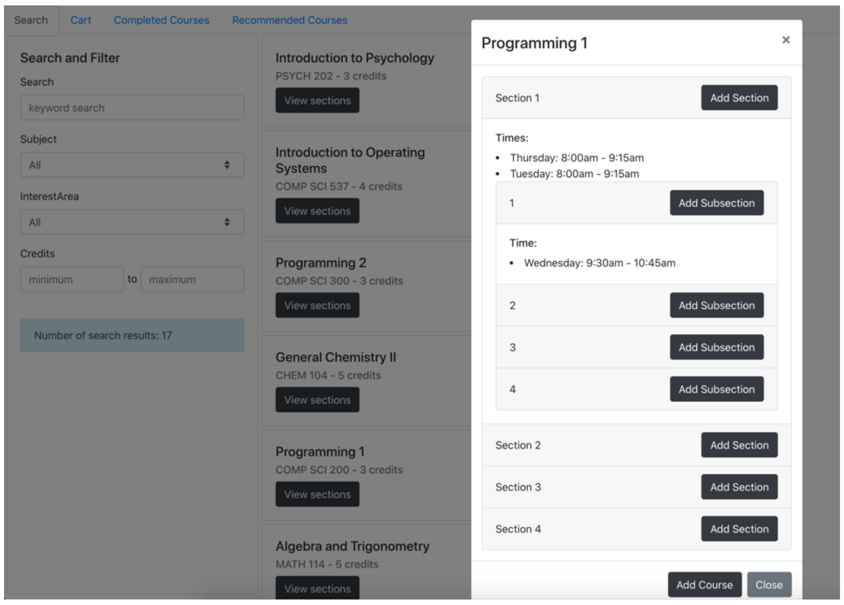
Programming 2
COMP SCI 300 - 3 credits
View sections

Programming 1
COMP SCI 200 - 3 credits
View sections

Introduction to Computer Engineering
COMP SCI 252 - 2 credits
View sections

Programming 3
COMP SCI 400 - 3 credits
View sections

Machine Organization and Programming



Search | Cart | Completed Courses | Recommended Courses

Search and Filter

Search

keyword search

Subject

All

InterestArea

All

Credits

minimum to maximum

Number of search results: 17

Introduction to Psychology
PSYCH 202 - 3 credits
View sections

Introduction to Operating Systems
COMP SCI 537 - 4 credits
View sections

Programming 2
COMP SCI 300 - 3 credits
View sections

General Chemistry II
CHEM 104 - 5 credits
View sections

Programming 1
COMP SCI 200 - 3 credits
View sections

Algebra and Trigonometry
MATH 114 - 5 credits
View sections

Programming 1 ×

Section 1 — Add Section

Times:
- Thursday: 8:00am - 9:15am
- Tuesday: 8:00am - 9:15am

1 — Add Subsection

Time:
- Wednesday: 9:30am - 10:45am

2 — Add Subsection

3 — Add Subsection

4 — Add Subsection

Section 2 — Add Section

Section 3 — Add Section

Section 4 — Add Section

Add Course | Close