|  |  |
| --- | --- |
| 206 | Work as part of a programming team, engaging in discussion with colleagues in ways that demonstrate an ability to understand other people’s code and offer constructive criticism. |

Asdsad

|  |  |
| --- | --- |
| 26 | Demonstrate the ability to describe user needs, and test how a design meets those needs. |

|  |  |
| --- | --- |
| 231 | Working as part of a group, take an open problem such as a simple design task, refine the problem in sensible ways and pose possible solutions. |

Asd

|  |  |
| --- | --- |
| 233 | Demonstrate a clear awareness of human interface issues, and why they are an important part of any design task. |

Asd

|  |  |
| --- | --- |
| 250 | Define functions to write the contents of a data structure to disk and read them back. |

Asdsada

|  |  |
| --- | --- |
| 255 | Write a function that uses stacks to solve a simple problem. |

Asdasd

|  |
| --- |
|  |

|  |  |
| --- | --- |
| 274 | Explain how a stack is used to evaluate expressions. |

<https://en.wikipedia.org/wiki/Stack_(abstract_data_type)>

<http://faculty.cs.niu.edu/~hutchins/csci241/eval.htm>

Stacks work as a physical stack of objects does - you can only add (push) or remove (pop) elements from the top.

For evaluating expressions, 3 types of notations can be used:

* Prefix – The type Racket uses, with the operation at the top:
  + Ex: (+ 1 2) = 3
* Infix – Operation in the middle (standard arithmetic notation):
  + Ex: 1 + 2 = 3
* Postfix – operation at the end
  + Ex: 1 1 + = 3

To evaluate those, stacks are used in different ways:

|  |  |
| --- | --- |
| 295 | For a given Racket program (either one provided or one you have written), produce a list of test cases to test it. Explain the choice of test cases. |

|  |  |
| --- | --- |
| 297 | Write pre- and post-conditions for simple Racket programs (either ones that you have written or ones provided). |

1. Pre-conditions are the things that must be true before a method is called. The method tells clients "this is what I expect from you".
2. Post-conditions are the things that must be true after the method is complete. The method tells clients "this is what I promise to do for you".

|  |  |
| --- | --- |
| 298 | Individually or in a group, demonstrate an understanding of top-down design. |

Make top-down for project

|  |  |
| --- | --- |
| 299 | Demonstrate an understanding of the difference between specification and implementation. |

|  |  |
| --- | --- |
| 300 | Discuss the respective advantages and disadvantages of different data structures, justifying choices for particular problems. |

|  |  |
| --- | --- |
| 308 | Build a simple network server using Racket |
| 309 | Parse a simple JSON file using Racket |

|  |  |
| --- | --- |
| 320 | Create and represent a Use Case |

<https://en.wikipedia.org/wiki/User_story>

<https://en.wikipedia.org/wiki/Extreme_programming>

<https://en.wikipedia.org/wiki/Use_case#Examples>

|  |  |
| --- | --- |
| 333 | Work effectively and timely as part of a group to present the results of a project. |

|  |  |
| --- | --- |
| 334 | Extend a GUI-based game in a nontrivial way. |

|  |  |
| --- | --- |
| 507 | Working individually or in a small group, find at least two aspects of Racket that have not been explicitly taught, research them and demonstrate the ability to use them in sensible ways. Fluency and a deep understanding are not expected, but independent study and a willingness to engage with the literature must be shown. |

|  |  |
| --- | --- |
| 535 | Take an open design task, refine the problem to a manageable scope, develop a robust working prototype, evaluate your solution and present this to a small group with clarity and authority. |

|  |  |
| --- | --- |
| 536 | Demonstrate an understanding of the way a graphical user interface works with Racket, and construct some simple examples to illustrate points. |

|  |  |
| --- | --- |
| 561 | Working individually or in a small group, explore the implementation of a Racket example implementation (for instance for a game). Provide a solution to substantially extend the chosen system. For example, you might provide a user interface. Where the work has been carried out in a group, each individual must be able to explain what is being presented. |

|  |  |
| --- | --- |
| 603 | For a given Racket program (either one provided or one you have written), implement test cases and assertions. |