|  |  |
| --- | --- |
| **Name: Marina Moshchenko** | **Date: TBA** |
| **Class: CS Programming** | **CS Standards:**  9-12.CT.1 Create a simple digital model that makes predictions of outcomes.  The focus is on using data to build alternative numerical models that can best represent a data set.  9-12.CT.6 Demonstrate how at least two classic algorithms work, and analyze the trade-offs related to two or more algorithms for completing the same task.  The focus of this standard is a high-level understanding that algorithms involve tradeoffs, especially related to memory use and speed. Students should understand that classic algorithms are solved problems that can be reused.  9-12.CT.8 Develop a program that effectively uses control structures in order to create a computer program for practical intent, personal expression, or to address a societal issue.  The focus is on combining different forms of repetition and conditionals, including conditionals with complex Boolean expressions.  9-12.CT.9 Systematically test and refine programs using a range of test cases, based on anticipating common errors and user behavior.  The emphasis is on perseverance and the ability to use different test cases on their programs and identify what issues are being tested in each case.  9-12.CT.10 Collaboratively design and develop a program or computational artifact for a specific audience and create documentation outlining implementation features to inform collaborators and users.  The focus is on the collaborative aspect of software development, as well as the importance of documenting the development process such that the reasons behind various development decisions can be understood by other software developers.  9-12.DL.2 Communicate and work collaboratively with others using digital tools to support individual learning and contribute to the learning of others.  Digital tools and methods should include both social and professional (those predominantly used in college and careers). Collaboration should occur in real time and asynchronously, and there should be opportunities for students to both seek and provide feedback on their thoughts and products. |
| **UNIT:**  **Algorithms** | **Period/Topic:**  Search Algorithms |

|  |
| --- |
| ***Focus (In the form of a question – How? or Why?):***  ***How do we implement the Linear and the binary Search Algorithms into programming?***  ***How the two search algorithms compare?*** |
| ***Learning Target(s) Explicitly Stated / Students will be able to (S.W.B.A.T.):***  Students will be able to identify, differentiate and implement Linear and Binary Search algorithms into programming. They will be able to compare the efficiency of them. |
| ***Academic Vocabulary:***  ***Search algorithms, linear search, binary search, iterative, efficiency, trace tables.*** |
| ***Start-Up Assessment For Learning (Determine Prior Knowledge):***  Part 1. Consider an array of consecutive numbers from 1 to 15. Develop a strategy for Number Guessing Game to guess the secret number chosen by computer in as less number of step as possible.  Part 2. In pairs: Try your strategy when playing the number guessing game with your programming partner. Describe your approach.  Part 3. Share out and discussion. 2-3 volunteer students share their approaches and students discuss the efficiency of these approaches. |

|  |  |
| --- | --- |
| ***Mini****-****Lesson + Guided Practice****:*  Part 1. Linear Search Algorithm.  (Smoothly transitioned from the warm up) | ***Notes/Questions/Answers:*** |
| **Group Activity 1** (small groups of 3-4):  a) Students write the code for Linear Search Algorithm using the material explained during the lesson and in [this Link](https://www.tutorialspoint.com/data_structures_algorithms/linear_search_algorithm.htm)[1].  b) Students use [this Link](https://www.khanacademy.org/computing/computer-science/algorithms/intro-to-algorithms/a/a-guessing-game)[3] to play the number guessing game with computer and read about linear and binary search algorithms. Student will play on a different scale/range of numbers (1-15 and 1-300). They count the number of guesses (red crosses) it takes them to guess computer’s secret number and compare the efficiency of the 2 algorithms. | public static int linearSearch (int[] array, int target){  int i;  for (i = 0; i < array.length; i++){  if (target == array[i]){  System.out.println("Linear: the seached item " + target + " has an index: " + i);  return i;  }  }  System.out.println("The searched number is not in the array (linear)");  return -1;  } |
| ***Mini****-****Lesson + Guided Practice****:*  Part 2. Binary Search Algorithm. |  |
| **Group Activity 2** (the same small groups of 3-4). a). Students develop a pseudocode for binary search algorithm using the material explained during the lesson and in [this Link](https://www.tutorialspoint.com/data_structures_algorithms/binary_search_algorithm.htm)[2].  b). Students start programming Binary Search algorithm using iterative approach. | public static int binarySearchIterative (int[] array, int target){  int left = 0;  int right = array.length - 1;    while (left <= right){  int mid = (left + right)/2;    if (target == array[mid]){ //just right, base case  System.out.println("Iterative binary: the seached item " + target + " has an index: " + mid);  return mid;  }else if (target > array[mid]){ //too low  left = mid +1;    }else { //(target < array[mid]), too high  right = mid - 1;  }  } //end of loop  System.out.println("The searched number is not in the array (binary iterative");  return -1;    } |

|  |
| --- |
| ***Exceeding Standards Task:***  Scale the program to have different inputs:  1-15, 1-300, 1-1000, 1-10,000, etc. Make a counter to count and display the number of steps required to complete each search. |
| ***At Standards Task:***  Complete all programming parts of the lesson to a properly functioning code. |
| ***Below Standards Task:***  If programming part can not be completed as expected, the pseudocode needs to be finished and commented. Resources [1] and [2] have pseudocode for both algorithms |

|  |  |
| --- | --- |
| ***Summary (What have you learned today?):***  Exit Ticket: 3-5 min  Consider the array of consecutive numbers 1-15.  1. Define one case where *Linear* Search algorithm *is more efficient* than the *Binary* Search Algorithm.  2. Define one case where *Binary* Search algorithm *is more efficient* than the *Linear* Search Algorithm.  3. Define one case where Linear Search algorithm *is equally efficient* to the Binary Search Algorithm. |  |

|  |
| --- |
| ***Homework:***   1. Finish programming if not completed in class. Review Linear and Binary search algorithms (all resources/links we used) and get ready for a closer look at these algorithms on the next class! 2. Make 2 trace tables - one for each search algorithm. Use 3 cases from today’s exit ticket to demonstrate each algorithm’s efficiency in each case.   Reminder – Exit Ticket: Consider the array of consecutive numbers 1-15.  1. Define one case where *Linear* Search algorithm *is more efficient* than the *Binary* Search Algorithm.  2. Define one case where *Binary* Search algorithm *is more efficient* than the *Linear* Search Algorithm.  3. Define one case where Linear Search algorithm *is equally efficient* to the Binary Search Algorithm. |

Resources:

[1] <https://www.tutorialspoint.com/data_structures_algorithms/linear_search_algorithm.htm> Linear search

[2] <https://www.tutorialspoint.com/data_structures_algorithms/binary_search_algorithm.htm> Binary Search

[3] <https://www.khanacademy.org/computing/computer-science/algorithms/intro-to-algorithms/a/a-guessing-game> Khan Academy Number Guessing Game

[4] <https://www.youtube.com/watch?v=sr_bR1WwcLY> Binary search step-by-step animation (video)