Content Objective: Students will use collections in Java to solve programmatic challenges.

|  |  |
| --- | --- |
| **On the Tech Horizon (10pts.)**  **link to a tech/coding related article or journal no more than one month old (no blogs or reddit clones see below)** | |
| URL: |  |
| Reaction/Commentary: |  |

|  |  |
| --- | --- |
| **Tech Terms and History (20pts.)**  **vocabulary from BJ p.311-p.378 and The Information Chapter 6 (definition/commentary/significance in your words)** | |
| Array syntax using new | int[]stuff = new int[2]; |
| Array syntax with initial values | int[] values = {5,5,3,1,8,9}; |
| Bounds Error | A bounds error occurs when trying to access an element of the array that doesn’t exist |
| Syntax to reference array element | array[index] |
| Syntax to copy array contents | Double[] array = Arrays.copyOf(values,n); |
| Partially filled arrays | A partially filled array has values that are unused, a companion variable should be used to keep track of how many of the elements are used |
| Syntax to initialize array with for loop | Int[] values = new int[] 20;  For(int I = 0; I <20; I ++){  Values[i]= I;  } |
| /Syntax for enhanced loop | For(double element: values){  Total = total + element;  }  total tracks the added element of the whole array, values is the name of the array |
| Syntax for sum and average | Double total = 0;  For(double element: values){  Total = total + element;  }  average = total/values.length; |
| Syntax for min and max | Double largest = values [0];  For (int I = 1; i< values.length; i++){  If(values[i]>largest){  Largest= values[i];  }  } |
| Syntax for element separators | For(int = 0;i<values.length;i++){  If(i>0){  System.out.println(“ | ”);//this is the separator  }  System.out.println()values[i];  } |
| Syntax for linear search | Int searchedValue = 25;  Int pos = 0;  Boolean found = false;  While(pos<values.length && !found){  If(values[pos] == searchedValue){  Found = true;  }else{  pos ++;  }  } |
| Syntax for removing elements | Values[pos] = values[currentSize -1];  Currentsize--;  If the array is ordered:  For(int I = pos+1; I < currentSize; i++){  Values[i-1]=values;  }  currentSize--; |
| Syntax for inserting elements | If(currentSize<values.length){  currentSize++  values[currentSize-1]= newElement;  }  for pushing into the middle of an array  if(currentSize<values.length){  currentSize++  for(int I = currentSize-1; i>pos; i--){  values[i] = values[i-1]  }  values[pos] = newElement  } |
| Syntax for swapping elements | Double temp = values[i];  Values[i]= values[j];  Values[j] = temp; |
| Syntax to copy array | Double[] prices = Arrays.copyOf(values, values.length); |
| Syntax for reading input | Scanner in = new Scanner(System.in);  while (in.hasNextDouble() && currentSize < values.length) {  values[currentSize] = in.nextDouble();  currentSize++; } |
| / Syntax Two dimensional array | Double[][] tableEntries = new double[rows][columns];  Or  Int[][] data = {  {16,4,2},  {6,3,2},  } |
| Syntax for accessing 2d element | valueAtFourThree = counts[4][3]; |
| Syntax for Accessing neighbor elements | [I (add to move down)][j (add to move right)] |
| Syntax for accessing rows and columns | BJ p.343 |
| 3D Array syntax | BJ p.347 |
| Syntax for ArrayList with new | BJ p.347 |
| Syntax for add, get, set, and remove | BJ p.350 |
| Regression testing | BJ p.357 |

|  |  |
| --- | --- |
| **Code Snippets (30pts.)**  **only submit snippets or classes no full programs required (test and run in IDE, then copy/paste applicable code frag)** | |
| Pseudocode Dice Roll Simulator with an Array (shows winner after 100 rolls) | Make an array with 200 integers, randomized 1-6, compare each number 1-100 with an offset in the array by 100, use a counter w to determine which dice won more |
| Actual Dice Roll Code | |  | | --- | | class D2{ | |  | public static void main(String[] args){ | |  | int w = 0; | |  | Random r = new Random(); | |  | int[] d = new int[200]; | |  | for(int i=0;i<200;i++){ | |  | d[i]=r.nextInt(6)+1; | |  | } | |  | for(int i=0;i<100;i++){ | |  | if(d[i] > d[i+100]){ | |  | w++; | |  | }else if(d[i] == d[i+100]){ | |  | }else{ | |  | w--; | |  | } | |  | } | |  | if(w>0){ | |  | System.out.print("1 wins"); | |  | }else if (w<0) { | |  | System.out.print("2 wins"); | |  | }else{ | |  | System.out.print("tie"); | |  | } | |  | } | |  | } | |
| Modify code to return first die of 100 wins |  |
|  |  |
| E7.2-E7.7 |  |
| E7.10-E7.13 |  |
| E7.20-E7.22 |  |

|  |  |
| --- | --- |
| **Code Challenge (30pts.)**  **full functioning application sent to GitHub** | |
| You may choose any of the following code P7.6-P7.15 | |
| Notes: |  |

|  |  |
| --- | --- |
| **Badge Progress (10pts.)**  **building your coding profile: Java coding training site to earn badges (recommended site** [**http://coderbyte.com**](http://coderbyte.com) **)** | |
| Screenshot/URL: |  |
| Notes/Issues: |  |

|  |  |
| --- | --- |
| **Notes**  **your notes** | |
| Notes: |  |