|  |  |
| --- | --- |
| **Page:** | 1  [2](https://takeexam.next.ecollege.com/(NEXT(1f0d9eda5e))/Main/CourseMode/StudentGradebookExam/StudentGradebookExamView.ed?examID=54417354&courseItemSubId=451821556&studentID=22698822&currentPageNumber=2&digest=L5dLDHCBBH%252fpp8Yj%252bXiRk%252fhH7gRdSsuosOcY%252fKUttvM%253d&) |

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | Question 1. | Question : | Describe the purpose of a Java class constructor. | |
| |  |  |  |  | | --- | --- | --- | --- | |  | Student Answer: |  | A Java constructor class defines the instructions that each class undergoes on its initialization process.  They are in essence the blueprint by which each class is constructed and created. | |  | | | | |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | |  | Points Received: | **0.6 of 0.6** | |  | Comments: |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | Question 2. | Question : | What is the purpose of set<attribute> and get<attribute> methods? | |
| |  |  |  |  | | --- | --- | --- | --- | |  | Student Answer: |  | Setters perform an action on a private field within a class this changes the field by the input value.  Getters return the value of the private field within a class.   The main purpose of these methods are to manipulate and extract data from private fields within a class. | |  | | | | |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | |  | Points Received: | **0.6 of 0.6** | |  | Comments: |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | Question 3. | Question : | Explain the term ***instance variable.*** | |
| |  |  |  |  | | --- | --- | --- | --- | |  | Student Answer: |  | An instance variable is a global variable, or a variable that is available to all instances within the program.  Most of the time these are variables that are created in one class and are available to all sub-classes to manipulate and reference. | |  | | | | |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | |  | Points Received: | **0.6 of 0.6** | |  | Comments: |  | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Question 4. | Question : | Given a Java int array containing the following values:   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 22 | 37 | 13 | 57 | 21 | -3 | 22 | 11 | 41 | -5 |  1. What is the order of the values following the first pass of the selection sort we used in class: 2. What is the order of the values following the second pass of the selection sort: 3. What is the order of the values following the third pass of the selection sort: 4. What is the order of the values following the fourth pass of the selection sort:  |  | | --- | |  | | |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  | Student Answer: |  | 1.   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | -5 | 37 | 13 | 57 | 21 | -3 | 22 | 11 | 41 | 22 |   2.   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | -5 | -3 | 13 | 57 | 21 | 37 | 22 | 11 | 41 | 22 |   3.   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | -5 | -3 | 11 | 57 | 21 | 37 | 22 | 13 | 41 | 22 |   4.   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | -5 | -3 | 11 | 13 | 21 | 37 | 22 | 57 | 41 | 22 |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | -5 | 37 | 13 | 57 | 21 | -3 | 22 | 11 | 41 | 22 | | -5 | 37 | 13 | 57 | 21 | -3 | 22 | 11 | 41 | 22 |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | -5 | 37 | 13 | 57 | 21 | -3 | 22 | 11 | 41 | 22 | | |  | | | | |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | |  | Points Received: | **3.6 of 3.6** | |  | Comments: |  | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Question 5. | Question : | Given a Java String array containing the following values:   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Alan | Bernard | Casper | Daniel | Dennis | Herb | Micheal | Stephen | Wilber | Zelda |  1. How many probes of the binary search are required to locate *Casper*? (explain how you reached this result) 2. How many probes of the binary search are required to determine *Waldo* is not in the list? (explain how you reached this result) | |
| |  |  |  |  | | --- | --- | --- | --- | |  | Student Answer: |  | My Definition of probes is any comparison of values within the search  1.  goto arraysize/2 == Dennis Is Dennis == Casper? No Count 1 Is Dennis < Casper? No Count 2 Eliminate all indexes > Dennis Newarraysize/2 == Casper Is Casper == Casper? Yes Count 3  3 Probes  2. goto arraysize/2 == Dennis Is Waldo == Dennis? No Count1 Is Waldo > Casper? Yes Count 2 Eliminate all indexes < Dennis Newarraysize/2 == Michael Is Michael == Waldo? No Count 3 Is Waldo > Michael? Yes Count 4 Eliminate all indexes < Michael Newarraysize/2 == Wilber Is Waldo == Wilber? No Count 5 Is Waldo > Wilber? No Count 6 Eliminate all indexes above Wilber Remaining Cells == Stephen Is Stephen == Waldo? No Count 7 Waldo is not found | |  | Instructor Explanation: | Each bit is 1.8 points. | | |  | | | | |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | |  | Points Received: | **3.6 of 3.6** | |  | Comments: |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | Question 6. | Question : | Given the following class for SailboatType:  public class SailboatType {      String Manufacturer;      String Model;      float Beam;      float Displacement;      float LOA;      float LWL;      public void setManufacturer (String manfct);      public String  getManufacturer ();      public void setModel (String model);      public String  getModel ();      public void setBeam (float beam);      public float getBeam ();      public void setDisplacement (float disp);      public float  getDisplacement ();      public void setLOA (float loa);      public float getLOA ();      public void setLWL (float lwl);      public float getLWL ();  }  How do you use it to implement a SailboatListType similar to the StringList programming problem? | |
| |  |  |  |  | | --- | --- | --- | --- | |  | Student Answer: |  | Create a SailboatListType class that modifies the this.list and stores the objects in a linked list style of program.  Than create a menu.java that creates the Sailboat objects and stores them in the SailboatListType so that you can reference each object's stored variables. There is no constructor for this object so I would probably create a constructor and have it initialize blank variables at least, or accept input from the menu.  Than have the objects created with the information and stored in the SailboatListType after modification.  i.e. Public class menu {      public static void main(String[] args){     SailboatType newobject1 = new SailboatType();     SailboatListType.add(newobject1);     SailboatType newibject2 = new SailboatType();     SailboatListType.add(newobject2);     etc. etc.     } } | |  | | | | |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | |  | Points Received: | **2.6 of 2.7** | |  | Comments: | More details on the list implementation or the implementation | | |

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | Question 7. | Question : | What is the characteristic behavior of a stack? | |
| |  |  |  |  | | --- | --- | --- | --- | |  | Student Answer: |  | A stack primarily operates on a First In Last Out methodology. In otherwords what you place into the stack, the next item will overtake the place in the stack. 1,2,3,4 Placed into a stack in that order would be only be able to extract 4,3,2,1 in that order. | |  | | | | |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | |  | Points Received: | **0.9 of 0.9** | |  | Comments: |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | Question 8. | Question : | What are the five basic operations on a stack? | |
| |  |  |  |  | | --- | --- | --- | --- | |  | Student Answer: |  | pop,push,peek,empty, search | |  | | | | |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | |  | Points Received: | **0.9 of 0.9** | |  | Comments: |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | Question 9. | Question : | Track stack myStack through the following operations (show what the stack looks like following each operation):  myStack.push (1)  myStack.push (5)  int topValue = myStack.pop ()  myStack.push (2)  myStack.push (9)  topValue = myStack.pop ()  topValue = myStack.pop ()  myStack.push (13)  myStack.push (8)  topValue = myStack.pop ()  topValue = myStack.pop ()  topValue = myStack.pop () | |
| |  |  |  |  | | --- | --- | --- | --- | |  | Student Answer: |  | myStack = "" myStack.push(1)    / myStack = {1} myStack.push(5)  / myStack = {5,1} int topValue = myStack.pop () / topValue = 5 / myStack = {1} myStack.push (2) / topValue = 5 /myStack = {2,1} myStack.push (9) / topValue = 5 / myStack = {9,2,1} topValue = myStack.pop () / topValue = 9/ myStack = {2,1} topValue = myStack.pop () / topValue = 2 /myStack = {1} myStack.push (13) / topValue = 2 /myStack = {13,1} myStack.push (8) /topValue = 2 /myStack = {8,13,1} topValue = myStack.pop () / topValue = 8 / myStack = {13,1} topValue = myStack.pop () /topValue = 13 / myStack = {1} topValue = myStack.pop () /topValue = 1 /myStack = {} | |  | | | | |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | |  | Points Received: | **2 of 2** | |  | Comments: |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | Question 10. | Question : | What is the difference between the stack and a queue? | |
| |  |  |  |  | | --- | --- | --- | --- | |  | Student Answer: |  | A stack will only reference the last object using a FILO methodology to extract data. While a queue will maintain the object's place in a queue and can operate in a FIFO and assign priority. | |  | | | | |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | |  | Points Received: | **0.9 of 0.9** | |  | Comments: |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | Question 11. | Question : | What are the basic operations on a queue? | |
| |  |  |  |  | | --- | --- | --- | --- | |  | Student Answer: |  | Add, offer, remove, poll, element, peek | |  | | | | |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | |  | Points Received: | **0.9 of 0.9** | |  | Comments: |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | Question 12. | Question : | Trace queue myQueue through the following operations (show what the queue looks like following each operation):  myQueue.enque (1)  int value = myQueue.deque ()  myQueue.enque (5)  myQueue.enque (13)  value = myQueue.deque ()  myQueue.enque (7)  myQueue.enque (21)  myQueue.enque (11)  value = myQueue.deque ()  value = myQueue.deque ()  value = myQueue.deque ()  myQueue.enque (4)  value = myQueue.deque ()  value = myQueue.deque ()  value = myQueue.deque () | |
| |  |  |  |  | | --- | --- | --- | --- | |  | Student Answer: |  | myQueue.enque (1) / myQueue = {1} int value = myQueue.deque () /value = 1/ myQueue = {} myQueue.enque (5) / value = 1/ myQueue = {5} myQueue.enque (13) / value = 1/ myQueue  = {5,13} value = myQueue.deque () / value = 5 / myQueue = {13} myQueue.enque (7) / value = 5/ myQueue = {13, 7} myQueue.enque (21) / value = 5/ myQueue  = {13,7,21} myQueue.enque (11) / value = 5/ myQueue = {13,7,21,11} value = myQueue.deque () /value = 13 / myQueue {7,21,11} value = myQueue.deque () / value = 7 / myQueue {21,11} value = myQueue.deque () / value = 21 / myQueue {11} myQueue.enque (4) / value = 21 / myQueue {11, 4} value = myQueue.deque () / value = 11 / myQueue {4} value = myQueue.deque () / value = 4 / myQueue {} value = myQueue.deque () / value = null / myQueue {} | |  | | | | |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | |  | Points Received: | **2.7 of 2.7** | |  | Comments: |  | | |