MID TERM SAMPLE SP ’18 (solutions)

1. Which of these is an incorrect array declaration?

**a) intarr[]=newint[5]** b) int [] arr = new int[5]

c) int arr[] = new int[5] **d) int arr[] = int [5] new**

2. What will this code print?

int arr[] = new int [5];

System.out.print(arr[0]);

**a) 0** b) value stored in arr[0] c) 00000 d) Class name@hashcode in hexadecimalform

3. What is the output of this program?

  class array\_output {

      public static void main(String args[])    {

          int array\_variable [] = new int[10];

   for (int i = 0; i < 10; ++i) {

              array\_variable[i] = i;

              System.out.print(array\_variable[i] + " ");

              i++;

          }

      }

  }

a) **0 2 4 6 8** b) 1 3 5 7 9 c) 0 1 2 3 4 5 6 7 8 9 d) 1 2 3 4 5 6 7 8 9 10

4. Look at the following code.

Integer myNumber = new Integer(5);

int var = myNumber;

What is true about the second statement?

a) The statement performs autoboxing.

b) It results in an error because you can’t assign a wrapper class to a primitive variable.

**c) The statement performs unboxing.**

d) The statement performs unwrapping.

5. Which is the valid declaration within an interface definition?

**a) public double methoda();** b) public final double methoda();

c) static void methoda(double d1); d) protected void methoda(double d1);

6. True or **False**? A private base method can be overridden by some derived public method.

7. Which of these keywords is not a part of exception handling?

a) try b) finally **c) thrown** d) catch

8. What line of code should replace the missing statement to make this program compile?

/\* Missing Statement ? \*/  
public class foo   
{  
   public static void main(String[]args)throws Exception   
   {  
       java.io.PrintWriter out = new java.io.PrintWriter();   
       new java.io.OutputStreamWriter(System.out);   
       out.println("Hello");   
   }   
}

* 1. **No statement required.** b) import java.io.\*;

c) include java.io.\*; d) import java.io.PrintWriter;

9. The process of converting one date type to another is called\_\_\_\_\_\_\_\_\_\_.

1. Translating    **b) Casting**c) Compiling    d) Declaring

10. Which of the following is the correct declaration statement in java program?

a) int num=int[5] b) int num=new num[5]

**c) int[] num=new int[5]** d) None

11. \_\_\_\_\_\_\_\_\_\_\_\_method is used to find the nth no. of character of given string s1.

1. 1.index(n) b) s1.substring(n) c) s1.length()  **d) s1.charAt(n)**

12. Which of these can be overloaded?

a) Methods b) Constructors **c) both a and b** d) None can be overloaded

13. The concept of multiple inheritance is implemented in Java by?

I.   Extending two or more classes.

II.  Extending one class and implementing one or more interfaces.

III. Implementing two or more interfaces.

a) Only (II) b) (I) and (II) **c) (II) and (III)** d) Only (I) e) Only (III)

14. True or **false**? Private members of a class are inherited in the sub class.

15. **True** or false? A method of a super class with a default access modifier can be overridden as protected or public but not as private.

16. What is the output of following linked list li?

ll.add("F");

ll.add("B");

ll.add("D");

ll.add("E");

ll.add("C");

ll.addLast("Z");

ll.addFirst("A");

ll.add(1, "A2");

**a) [A, A2, F, B, D, E, C, Z]** b) [ Z, C, E, D, B, F, A2, A]

c) [F, B, F, D, E, C, Z, A, 1 A2] d) [ A, 1 A2, F, B, D, E, C, Z]

17. What will be the output of the following program?

class X {

void method(int a) {

       System.out.println("ONE");

 }

 void method(double d)  {

       System.out.println("TWO");

 }

}

class Y extends X {

  @Override

 void method(double d) {

       System.out.println("THREE");

  }

}

public class MainClass {

   public static void main(String[] args)  {

       new Y().method(100);

   }

}

1. **ONE**  b) TWO c) THREE d) ONE TWO e) Error

18. If a superclass does not have a default constructor,

**a) then a class that inherits from it, must call one of the constructors that the superclass does have**

b) then a class that inherits from it, must contain the default constructor for the superclass

c) then a class that inherits from it, does not inherit the data member fields from the superclass

d) then a class that inherits from it, must initialize the superclass values

19. The binary search algorithm

**a) will cut the portion of the array being searched in half each time the loop fails to locate the search value**

b) will have an average of N/2 comparisons, where N is the number of elements in the array

c) is less efficient than the sequential search algorithm

20. Which method correctly measures time to a billionth of a second.

a) nanoTimer() b) milliseconds() c) microseconds() **d) nanoTime()**

Coding and Definitions:

21. What are some differences between an ArrayList and a LinkedList?

Basically **LinkedList is faster in add and remove, but slower in get.**

Proof:

**-get(int index): ArrayList gives the performance of O(1) while LinkedList performance is O(n)**

\*ArrayList maintains **index** based system for its elements as it uses array data structure implicitly which makes it faster for searching an element in the list. On the other side LinkedList implements doubly linked list which requires the **traversal** through all the elements for searching an element.

removals: **LinkedList remove operation gives O(1**) performance while **ArrayList gives variable performance: O(n) in worst case (while removing first element) , O(1) in best case (while removing last element).**

\***LinkedList’s each element maintains two pointers (addresses)** which p**oints** to the both neighbor elements in the list. Hence removal only requires change in the pointer location in the **two neighbor nodes (elements) of the** node which is going to be removed. With **ArrayList all the elements need to be shifted** to **fill** out the space created by removed element.

adds: Performance: **LinkedList add method gives O(1) performance while ArrayList gives O(n) in worst case.** Reason is same as that for removals.

Final note. Memory Overhead: ArrayList maintains indexes and element data while LinkedList maintains element data and two pointers for neighbor nodes hence memory consumption is high in LinkedList in comparison.

22. What is the basic difference between the Comparable and Comparator interfaces?

Comparable interface uses **public int compareTo(Object o)**

Compares this **Object** **o** with the specified object for order. Returns a negative integer,  zero, or a positive integer as this object is less than, equal to, or greater than the  specified object. Good for single field sorts.

Comparator interface uses **public int compare (Object o1,Object o2)**  
Compares its two arguments for order. Returns a negative integer, zero, or a positive  integer as the first argument is less than, equal to, or greater than the second. Good for multiple field sorts.

23. What are some difference between Queue and Stack? How are implementation performed for each?

Queue is FIFO implementation, stack is LIFO.

LIFO: object item is inserted last and is retrieved first.

FIFO: object item is first inserted, and is first consumed, as insertion and consumption happen at the opposite ends of the queue.

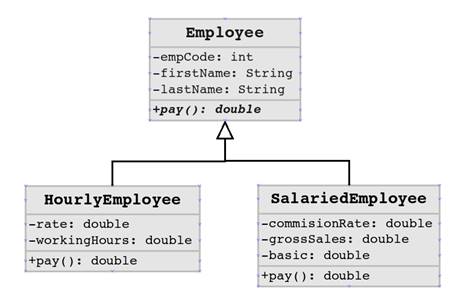
24. What is the major difference between an abstract class and an interface?

Abstract class is more *expensive* to use, as there is a look-up to do when you inherit from them. Can define a behavior for common classes.

An interface is a contract, a generic pattern of methods with just signatures allowing any class implement its methods in a way that it wishes to do.

25. Define Polymorphism? What are some of its abilities/usefulness?

Means a state of having many shapes or the capacity to take on various forms. Ability to describe and process objects of various types and classes through a single, uniform interface (base object).



26. Name a minimum of 4 data types (2 primitive and 2 reference types) supported in java?

int, double, Integer, Double

27. Code the sort method below to sort an arraylist in ascending order? (DO NOT USE ANY PRE DEFINED METHODS)

import java.util.ArrayList;

public class aListClass{

        public static void main(String[] args) {

        ArrayList<Integer> values = new ArrayList<>();

        values.add(1);

        values.add(41);

        values.add(3);

        sort(values);

        for (int i = 0; i < values.size(); i++)

            System.out.println(values.get(i));

    }

    public static void sort(ArrayList<Integer> aList) {

       int temp;

        for (int i = 0; i < aList.size() - 1; i++) {

            for (int j = i + 1; j < aList.size(); j++) {

                if (aList.get(i) > (aList.get(j))) {

                    temp = aList.get(i);

                    aList.set(i, aList.get(j)); //swap

                    aList.set(j, temp);         //values

                }

            }

        }

    }

}

28. Tracing the code! You have a queue with 10 integers Q = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 }. The expected output is Q = { 1, 6, 2, 7, 3, 8, 4, 9, 5, 10 }. Fill in any blanks alongside comments where shown to show trace results of the stack or queue.

**public** **class** InterleavingElements {

**public** **static** **void** main(String[] args) {

Queue<Integer> q = **new** LinkedList<Integer>();

//Adding integers from 1 to 10

**for** (**int** i = 1; i <= 10; i++) {

q.add(i); //**1,2,3,4,5,6,7,8,9,10**

}

InterleavingElements ii = **new** InterleavingElements();

System.***out***.println("this is queue " + q.toString());

ii.InterchangingInterleaving(q);

System.***out***.println("this is queue " + q.toString());

}

**public** Queue<Integer> InterchangingInterleaving(Queue<Integer> q){

Stack<Integer> st = **new** Stack<Integer>();

**int** size = q.size()/2;

**for**(**int** i = 1; i <= size; i++){

st.push(q.remove()); //**1,2,3,4,5**

}

**while**(!st.isEmpty()){

q.add(st.pop()); //**6,7,8,9.10,5,4,3,2,1**

}

**for**(**int** i = 1; i <= size; i++){

q.add(q.remove()); //**5,4,3,2,1,6,7,8,9,10**

}

**for**(**int** i = 1; i <= size; i++){

st.push(q.remove()); //**5,4,3,2,1**

}

**while**(!st.isEmpty()){

q.add(st.pop()); //**1, 2, 3, 4, 5**

q.add(q.remove()); //6, 7, 8, 9, 10

}

**return** q;

}

29. Create and define a method called **popList** that gets passed an **arraylist** called list that will accept any numeric value type passed in. The method definition should assign 5 integers to the arraylist via a loop storing the values of 1,2,3,4 and 5 respectively.

static void popList (ArrayList<Number> list int num) {

for( int i=1; i<=num; ++i)

list.add(i);

}

30. A challenge! Finish the code below to draw a bar chart as displayed below, given the following **grades** array and **counter** array which holds the grade range totals being primarily in groups of 10.

00-09:

10-19:

20-29:

30-39:

40-49:

50-59:

60-69: \*\*

70-79: \*\*\*

80-89: \*\*

90-99: \*\*

100: \*

**int** grades[] = { 60, 70, 75, 100, 61, 83, 90, 89, 72, 91 };

**int** counter[] = **new** **int**[grades.length + 1];

**for** (**int** i = 0; i < grades.length; ++i) { // loop over entire array

// inspect each element and increment the index

// if the grade is in a certain range (ex. 0-9, 10-19…)

**if** (grades[i] == 100) {// account for the odd ball

counter[10]++;

**else**

counter[(grades[i] % 100) / 10]++;

}

// show the bar chart per range above

**int** x=0;

String stars = "";

**while** ( x < counter.length) {

**if** (x==10)

System.***out***.printf("%5d: ",100);

**else**

System.***out***.printf("%02d-%02d: ",x\*10,x\*10+9);

**for** (**int** i = 0; i < counter[x]; ++i) {

stars = "\*";

System.***out***.print( stars );

}

System.***out***.println();

stars="";

x++;

}