Data Structures and Algorithms Laboratory | **DES-ITS 231**
Examination #1 Intro to Java For/While loop [Version 6]
Name Krittanut Yongsuphanomgko **ID** 6522770542 **Section** ?

Score 6/6

Please complete the following Java programs to generate the expected console output.

Problem 1: Printing Fibonacci Sequence

This program prints the Fibonacci sequence up to a given number 'n' ***using while loop***. The Fibonacci sequence is a series of numbers where each is the sum of the two preceding ones, typically beginning with 0 and 1.

```
public class Fibonacci {
  private static void printFibonacciSequence(int n) {
    int firstTerm = 0;
    int secondTerm = 1;
    int i = 1;
    while (i <= n)                                  { //Q1
      System.out.print(firstTerm + " ");
      int nextTerm = firstTerm + secondTerm;
      firstTerm = secondTerm                        ; //Q2
      secondTerm = nextTerm;
      i++                                           ; //Q3
    }
  }

  public static void main(String[] args) {
    printFibonacciSequence(10);
  }
}
```

(1) (1) (1)

This is the console output of the program when it is executed.

1 2 3 4 5 6 7 8 9 10

```
0 1 1 2 3 5 8 13 21 34
```

Problem 2: Printing a Sum Table

This program prints a sum table. The sum table displays each cell's sum of row and column indices using ***nested for loop***. Please complete the code and achieve the desired output.

```
public class SumTable {
  private static void printSumTable(int m, int n) {
    for(int i=0; i < m; i++)                        { //Q4
      for (int j = 0; j < n; j++) {
        System.out.print((i+j)+" ")                 ; //Q5
      }
      System.out.println();
    }
  }

  public static void main(String[] args) {
    printSumTable(6, 6)                             ; //Q6
  }
}
```

(1) (1) (1)

six five

This is the console output of the program when it is executed.

```
0 1 2 3 4
1 2 3 4 5
2 3 4 5 6
3 4 5 6 7
4 5 6 7 8
5 6 7 8 9
```

0,1,2,3,4
0,1,2,3,4,5

Seint

Data Structures and Algorithms Laboratory | DES-ITS 231
Examination #2 Recursion and Array [Version 6]
Name Krittanut Yongsuphamongkol **ID** 6522770542 **Section** S

Score: 6

Please complete the following Java methods and programs by filling in the blanks.

Problem 1

Please implement the integer array's `removeAtIndex(int index)` and `addLast(int e)` methods. The removeAtIndex(int index) method of an array is an operation that removes an element from the array at a specified index, shifting all subsequent elements to fill the gap created by the removal. The `addLast()` method appends or adds an element to the end of the array.

```
int removeAtIndex(int index) {
   if (load > 0) {
      int temp = arr[index];

      for (int i = index; i < load - 1 ; i++) {   // Q1
         int rightIndex = i + 1;
         if (rightIndex < size) {
            arr[i] = arr[rightIndex];
         }
      }
      arr[load - 1] = 0;   // Q2
      load --;   // Q3
      return temp;
   } else {
      return 0;
   }
}

void addLast(int e) {
   if (load < size) {
      arr[load] = e;   // Q4
      load++;
   }
}
```

Array
size: int load: int arr: int[]
addFirst(e: int): void addLast(e: int): void addAtIndex(e: int, index: int): void removeFirst(): int removeLast(): int removeAtIndex(index: int): int getElementAtIndex(index: int): int setElementAtIndex(e: int, index: int): void printArray(): void +main(args: String[]): void

Problem 2: Recursively calculates the sum of odd numbers

Using recursion, this Java program calculates the sum of odd numbers from a given positive integer from n to 1. It performs this computation using a recursive function called `sumOfOdds(int n)`.

```
static int sumOfOdds(int n) {
   if (n <= 0) {
      return 0;   // Q5
   } else {
      if (n % 2 != 0) {
         return n + sumOfOdds(n-1);   // Q6
      } else {
         return sumOfOdds(n - 1);
      }
   }
}
```

This is the console output of the program when `sumOfOdds(10)` *is executed from the* `'main'` *method.*

```
25
```

NYAN

SIIT SINCE 1992

Data Structures and Algorithms Laboratory | DES-ITS 231
Examination #3 Sorting and Hashing [Version 6]
Name Krittamut Yongsuphamongkol ID 6822770542 Section 5 .

Score 6/6

Please complete the following Java methods and programs by filling in the blanks. (10 Minutes) TA Tomat

Problem 1: Search a hash table that was previously filled with the folding hash function, and the collision was solved using quadratic probing.

Good job

```
int searchHashTableFoldingQuadratic(int key) {
   int C = 100;
   int orig_addr = (key % C + key / C) % hSize;
   int fi = 0 // or int addr = orig_addr ; // Q1
   int i = 0;
   while (H[addr] != key) {
      fi = i*i ; // Q2
      addr = (orig_addr + fi) % hSize; // Q
      i++ ; // Q3
   }
   return addr;
}
```

HashTable
H: int[] hSize: int
fillHashTableFoldingQuadratic (key: int): void searchHashTableFoldingQuadratic (key: int): int

Problem 2: Quick Sort

```
static void RecursiveQuickSort(int[] A, int start, int end) {
  if (start < end) {
     int pivot = A[end];
     int[] L = new int[A.length];
     int[] R = new int[A.length];
     int l_count = 0;
     int r_count = 0;
     for (int i = start; i < end; i++) {
       if (A[i] < pivot) {
          L[l_count] = A[i];
          l_count++;
       } else {
          R[r_count] = A[i];
          r_count++;
       }
     }

     for (int i = 0; i < l_count; i++) {
        A[start + i] = L[i] ; // Q4
     }
     A[start + l_count ] = pivot ; // Q5
     for (int i = 0; i < r_count; i++) {
       A[l_count + start + i + 1] = R[i];
     }
     RecursiveQuickSort(A, start, start + l_count - 1);
     RecursiveQuickSort(A, start + l_count + 1, start + l_count + r_count) ; // Q6
  }
}
```

G1

SIIT SINCE 1992

Data Structures and Algorithms Laboratory | DES-ITS 231
Examination #4 Slist [Version 6]
Name Krittamet Kongsuphamongkul **ID** 6622770542 **Section** 5.

Score 5.5/6

Please complete the following Java methods and programs by filling in the blanks.

Q1-Q5 Fill the result after running that line; Q6-Q8 Fill in the blank.

```
public class SlistV6 {
  public static void main(String[] args){
  SList<Integer> ilist = new SList<Integer>();
  ilist.addFirst(3);
  ilist.addLast(6);
  ilist.addFirst(3);
  ilist.printHorizontal();
  ilist.addAtIndex(4, 1);
  ilist.addLast(7);
  ilist.addAtIndex(1, 4);
  ilist.printHorizontal();
  System.out.println(ilist.removeAtIndex (5));
  System.out.println(ilist.removeFirst ());
  ilist.printHorizontal();
  }
}
```

3, 3, 6

3 3 6 / Q1 (0.5 point) 0.5

3, 4, 3, 6

3, 4, 3, 6, 7

3, 4, 3, 6, 1, 7

3 4 3 6 1 7 / Q2 (1 point) 1 0.5

7 / Q3 (0.5 point)

3 / Q4 (0.5 point) 0.5

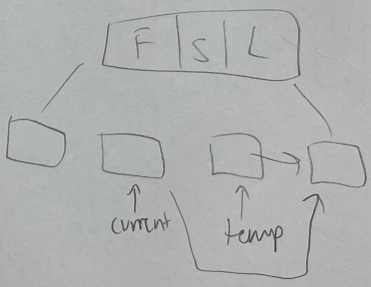
4 3 6 1 / Q5 (1 point) 1

```
//Class methods
  T removeAtIndex(int index){
    if(index < 0 || index > size)
      return removeLast(); /Q6 (0.5 point)
    else if(index==0)
      return removeFirst();
    else if(index==size-1)
      return removeLast();
    else {
      Node<T> current = first;
      for(int i=0; i < index-1; i++)
        current=current.next;

      Node<T> tmp = current.next /Q7 (1 point)

      current.next = tmp.next /Q8 (1 point)
      tmp.next=null;
      size--;
      return tmp.element;
  }
```

return null? ✗ 0



NYAN

G1

SIIT SINCE 1992

Data Structures and Algorithms Laboratory | DES-ITS 231
Examination #5 Stack and Application [Version 6]
Name Krittanut Tongsuphamongkol ID 6522770542 Section 5

Score 6/6

Please complete the following Java methods and programs by filling in the blanks.

```
static boolean isPalindrome(String word) {
Stack<Character> S1 = new Stack<Character>();
for (int i=0; i < word.length(); i++) {                    /Q1 (1)
    S1.push(word.AtChar[i]);                               /Q2 (1)
Stack<Character> S2 = S1.reverseStack();
while (!S1.isEmpty()) {
    if (S1.pop() != S2.pop()) {                            /Q3 (1)
        return false; }
}
return true;
}
```

word = word.toLowerCase(); ? 100%

or (!S1.pop().equals(S2.pop()))

```
void push(T element) {
    list.addFirst(T);                                      /Q4(0.5)  0.25
}
T pop() {
    return list.removeFirst();                             /Q5(0.5)  0.5
}
```

element

```
public class StackV6 {
  public static void main(String[] args){
    Stack<String> number = new Stack<String>();            /Q6 (1)
    number.push("77");
    number.pop();                                          /Q7 (0.5)  0.5
    number.push("12");
    System.out.println("Stack:"+number);
    number.push("13");                                     /Q8 (0.5)  0.5
    System.out.println("Stack:"+number);
    }
}
```

```
Stack:[12]
Stack:[12, 13]
```

NYAN

G1

SIIT SINCE 1992

Data Structures and Algorithms Laboratory | **DES-ITS 231**
Examination #6 Queue and Applications [Version 6]
Name Kritanut Yongsuphmongkol **ID** 6522770542 **Section** 5.

Score 6/6

Please complete the following Java methods and programs by filling in the blanks. (15 minutes)

```
97⊖   static void makeRoundRobin(Queue<Integer> Q, Queue<String> P, int limit, int resourceAmt)
98    {
99      printRoundRobin(Q, P, resourceAmt);
100
101     while(! Q.isEmpty() && resourceAmt!=0)
102     {    int temp =Q.dequeue();
103          String name = P.dequeue();
104          if( limit<=resourceAmt)
105          {  if(temp >= limit)
106            {
107
108             if(temp>0)
109             {   Q.enqueue(temp);
110                 P.enqueue(name);
111             }
112             printRoundRobin(Q, P, resourceAmt);
113            }
114          else
115          {
116              printRoundRobin(Q, P, resourceAmt);
117           }
118        }
119       else{  if(temp>resourceAmt)
120              {  temp=temp-resourceAmt;
121                 resourceAmt=0;
122                Q.enqueue(temp);
123                P.enqueue(name);
124              }
125              else
126              { resourceAmt=resourceAmt-temp;
127                temp=0;
128              }
129
130            printRoundRobin(Q, P, resourceAmt);
131     } //end if
132    }//end while
133   }
```

```
151   Queue<Integer> RRB = new Queue<Integer>();
152      RRB.enqueue(6);
153      RRB.enqueue(12);
154      RRB.enqueue(4);
155      RRB.enqueue(5);
156
157      RRB.enqueue(9);
158      Queue<String> P = new Queue<String>();
159      P.enqueue("P1");
160      P.enqueue("P2");
161      P.enqueue("P3");
162      P.enqueue("P4");
163      P.enqueue("P5");
164      P.enqueue("P6");
165      int amount = 40;
166      int quota = 3;
167      System.out.println("Your result is ");
168      QueueApp.makeRoundRobin(RRB, P, quota, amount );
169
170      System.out.println("The correct result should be ");
171      System.out.println(
172     "40: P1-6 P2-12 P3-4 P4-5 P5-7 P6-9 \n" +
173     "37: P2-12 P3-4 P4-5 P5-7 P6-9 P1-3 \n" +
174     "34: P3-4 P4-5 P5-7 P6-9 P1-3 P2-9 \n" +
175
176     "28: P5-7 P6-9 P1-3 P2-9 P3-1 P4-2 \n" +
177     "25: P6-9 P1-3 P2-9 P3-1 P4-2 P5-4 \n" +
178     "22: P1-3 P2-9 P3-1 P4-2 P5-4 P6-6 \n" +
179     "19: P2-9 P3-1 P4-2 P5-4 P6-6 \n" +
180     "16: P3-1 P4-2 P5-4 P6-6 P2-6 \n" +
181     "15: P4-2 P5-4 P6-6 P2-6 \n" +
182     "13: P5-4 P6-6 P2-6 \n" +
183     "10: P6-6 P2-6 P5-1 \n" +
184     "7: P2-6 P5-1 P6-3 \n" +
185     "4: P5-1 P6-3 P2-3 \n" +
186     "3: P6-3 P2-3 \n" +
187     "0: P2-3 ");
```

Q1 From Line No. 173 , the number 37 is calculated from source code line No? Complete the source code.

Answer 106: resourceAmt -= limit; (1) (0.5+0.5point)

Q2 P1-3 from Line No. 173 , the number 3 is calculated from source code line No?? Complete the source code.

Answer 107: temp -= limit; (1) (0.5+0.5point)

Q3 From Line No. 178➔179 , Why P1 is disappeared from queue? Which condition or which line made P1 disappear? Please explain. (1)

Answer
in the while loop when P1-3 it satisfy the if statement in line 104 and 105 then in line 107 temp is now 0 therefore the line 108 doesn't run therefore it didn't get enqueue back. (1 point)

Q4 From Line No. 180➔181 , Why the first number of row change from 16➔15? What line No. of source code used to calculate this new value? Complete the source code.

Answer Line 115: resourceAmt -= temp; (1) (0.5+0.5point)

Q5 Complete source code of Line No. 156 ? Answer RRB.enqueue(7); (1 point) (1)

Q6 Complete source code of Line No.175 (1)

"31: P4-5 P5-7 P6-9 P1-3 P2-9 P3-1 \n" + NYAN