السؤال 1

public class Node {  
  
 private int self;  
private int node1;  
  
 public int getSelf() {  
 return self;  
 }  
  
 public void setSelf(int self) {  
 this.self = self;  
 }  
 node1 = Node(5)  
 node1 = Node(10)  
 node1 = Node(15)  
   
 node1.next = node2  
 node2.next = node3

السؤال 2

public class circularlylinkedlist {  
 private int self;  
 private int current;  
 private int count;  
 private int None;  
 def is\_empty(self);  
 return self.head = None  
  
 def size(self):  
 count=0  
 while current !=;  
 count += 1;  
 current = current;  
  
 def prepend(self.data);

السؤال 3

public class m3 {  
 private int def;  
 private int self;  
 def is\_empty(self):  
 return self.size{  
   
 def add last(self):  
 if self.is\_empty():  
 self.head = new node  
 self.tail = new node  
 new \_node.next = new node   
   
 def remove\_frist(self):  
 if self.is\_empty():  
 return None  
   
 removed\_data=self.head.next;  
 self.size -=1;  
 return remove\_data;  
   
 def rotate(self):  
 if self.size>  
 }

السؤال 4

public class m4 {  
 public void addFirst(E e);  
 Node<>(e);  
 if(isEmpty())  
 private final java.lang.Object newest;  
  
 {  
 newest.setNext(newest);  
 tail=newest;  
 { else}  
   
 newest.setNext(tail.getNext());  
 tail.setNext(newest);  
 }  
 size++;  
}

السؤال 5

public class m5 {  
 def is\_empty(self);  
 private int self;  
 return self.head = is None;  
   
   
 class def {  
  
 def append(self, data):  
private int data;   
 private int count;  
 private int current;  
 else:  
 current =self.head;  
   
 while current.next !=self.head  
   
 def size(self):  
 if self.is\_empty():  
 return 0  
 count 1;  
 current = self.head.next;  
 while current !=self.head;  
count +=1;  
current = current.next  
 return count

السؤال 6

public class m6 {  
  
   
 public <objectobj, size> boolean equals(objectobj) {  
 return false;  
 if (this == objectobj) {  
 return true;  
  
   
 Node<E> walkA =tail.getNext();  
Node<?> walkB =other.tail.getNext();  
  
while (walkA != tail);  
if (!walkA.getElement);  
return false;  
 }  
 return walkA.getElement;  
 equals(walkB.getElement());  
 }  
}

السؤال 7

class m7 {  
 def comareCircularlists(L,M):  
 currentL = L.head  
 currentM =M.head  
   
 while currentL is not None:  
 if currentL.data != currentM.data;  
 return false;  
   
 current = currentL.next;  
 currentM = currentM.next;  
   
 if currentL == L.head:  
 return true;  
   
 if L.head is None and M.head  
 is None:  
   
 return true;  
 return false;  
  
}

السؤال 8

تحتوي L على عدد زوجي من العقد إلى قائمتين مرتبطتين دائرية بنصف الحجم يمكنك اتباع الخطوات التالية:

1. ابحث عن نقطة المنتصف للقائمة المرتبطة الدائرية باستخدام تقنية المؤشرين ابدا بمؤشرين بطيء وسريع وكلاهما يشير في البداية إلى رأس القائمة.
2. قم بتعين رأس القائمة المرتبطة دائرية الأولى للقائمة الأصليةL.
3. قم بتعين رأس القائمة المرتبطة دائرية الثانية للقائمة 2 باعتبارها العقدة البطيئة التالية (عقدة نقطة المنتصف).
4. قم بأنشاء العقدة التالية للنقطة البطيئة في راس القائمة 1 مما يودي إلى اغلاق القائمة 2 بشكل فعال في شكل دائري مرتبط.

/\*/  
original circular linked list:  
1-> 2-> 3-> 4-> 5-> 6-> 1  
(size =6)  
  
After splitting:  
  
list1 (size =3):  
1-> 2-> 3-> 1  
  
list2 (size =3):  
4-> 5-> 6-> 4  
  
 \*/

السؤال 9

public class m9 {

private int current;  
private int self;  
  
private int data;  
 class def {  
 def is\_empty(self);  
 return self.head = is None;  
 def add(self,data):  
 if self.is\_empty():  
 else:  
 current =self.head;  
 while current.next !=self.head  
def display (self):  
 if is\_empty();  
 print("circularlylinkedlist is empty.");  
 else  
 current = self.head  
 while true:  
 print(current.data,and="")  
 current=current.next;  
  
 }  
}