**Set A**

| **Python** | **Java** |
| --- | --- |
| def sum\_stack(st):  count=0  new\_stack= Stack()  while st.isEmpty() ==False:  new\_stack.push(st.pop())  count+=1  mid=count//2  i=0  temp= Stack()  while new\_stack.isEmpty() ==False:  if(i<mid):  temp.push(new\_stack.pop())  else:  st.push(temp.pop()+new\_stack.pop())  i+=1  return st | public static Stack sum\_stack(Stack stack) {  int count = 0;  Stack new\_stack = new Stack();  while (!stack.isEmpty()) {  new\_stack.push(stack.pop());  count++;  }  int mid = count / 2;  int i = 0;  Stack temp = new Stack();  while (!new\_stack.isEmpty()) {  if (i < mid) {  temp.push(new\_stack.pop());  } else {  stack.push(temp.pop()+new\_stack.pop());  }  i++;  }  return stack;  } |

**Set B**

| **Python** | **Java** |
| --- | --- |
| def sub\_stack(st):  count=0  new\_stack= Stack()  while st.isEmpty() ==False:  new\_stack.push(st.pop())  count+=1  mid=count//2  i=0  temp= Stack()  while new\_stack.isEmpty() ==False:  if(i<mid):  temp.push(new\_stack.pop())  else:  st.push(new\_stack.pop()-temp.pop())  i+=1  return st | public static Stack sub\_stack(Stack stack) {  int count = 0;  Stack new\_stack = new Stack();  while (!stack.isEmpty()) {  new\_stack.push(stack.pop());  count++;  }  int mid = count / 2;  int i = 0;  Stack temp = new Stack();  while (!new\_stack.isEmpty()) {  if (i < mid) {  temp.push(new\_stack.pop());  } else {  stack.push(new\_stack.pop() - temp.pop());  }  i++;  }  return stack;  } |

**RUBRIC**

|  | **Criteria** | **Marks** |
| --- | --- | --- |
| **1** | **Properly declaring method/function using proper parameter** | **1** |
| **2** | **Calculating the length of the stack** | **3** |
| **3** | **Calculate the mid of the stack** | **2** |
| **4** | **Store elements before mid in a temp stack** | **3** |
| **5** | **add/subtract the rest of the elements from the temp stack elements** | **3** |
| **6** | **Push the added/subtracted result in ht original stack** | **2** |
| **7** | **Returning the original stack** | **1** |

Note\*: There are multiple ways to solve this problem, and appropriate marks can be given for each approach based on its correctness and efficiency.