A logo of a university

Description automatically generated

NAMA : RAKASYAEL HIZKIA KOLONDAM

NIM : 2023105490

MATA KULIAH : TI0091 – PRAKTIKUM DATA TERSTRUKTUR

PRAKTIKUM KE - : 5

|  |  |  |
| --- | --- | --- |
| **No.** | **Kode** | **Penjelasan** |
| 1 | public Queue(int value)  {  Node newNode = new Node(value);  first = newNode;  last = newNode;  length = 1;  } | Ini adalah konstruktor untuk assign value ketika ingin menambahkan node baru ke dalam queue tanpa harus di panggil dua kali. |
| 2 | public void printQueue()  {  Node temp = first;  while (temp != null)  {  System.out.print(temp.value+" ");  temp = temp.next;  }  System.out.println();  } | Ini adalah kode untuk menampilkan antrian dengan cara iterasi setiap node yang ada di dalam queue sehingga selesai. |
| 3 | public void getFirst()  {  if (first == null)  {  System.out.println("First: null");  }  else  {  System.out.println("First: " +first.value);  }  } | Kode untuk menampilkan kepala dari queue atau node paling depan. Dilakukan pengecekan jika list kosong maka akan menampilkan null dan jika tidak kosong maka akan menampilkan node |
| 4 | public void getLast()  {  if (last == null)  {  System.out.println("Last: null");  }  else  {  System.out.println("Last: " + last.value);  }  } | Kode untuk menampilkan node paling belakang dari queue. Dilakukan pengeecekan jika list kosong maka akan menampilkan null, jika tidak akan menampilkan node terakhi. |
| 5 | public void getLength()  {  System.out.println("Length: " + length);  } | Metode untuk menampilkan panjang dari queue setelah di modifikasi. |
| 5 | public void enqueue(int *value*) {  Node newNode = new Node(*value*);   if(first == null){  first = newNode;  last = newNode;  }  else{  last.next = newNode;  last = newNode;  }   length++; } | Metode untuk menambahkan node dibelakang atau dequeue. Dilakukan pengecekan jika list kosong maka first dan last adalah node baru, jika tidak maka akan membuat node terakhir adalah node terbaru. Menunjuk node setelah last adalah node baru dan assign node baru menjadi last. |
| 6 | public int dequeue() {  if(first == null){  System.*out*.println("Empty List");  length--;  return -1;  }  else{  Node current = first;  first = first.next;  current.next = null;  length--;  return current.value;  } } | Menghapus node paling depan dengan mereturn integer yaitu value dari node yang dihapuskan. First akan digantukan dengan node selanjutnya lalu node setelah current akan menjadi null dan mengembalikan value. |