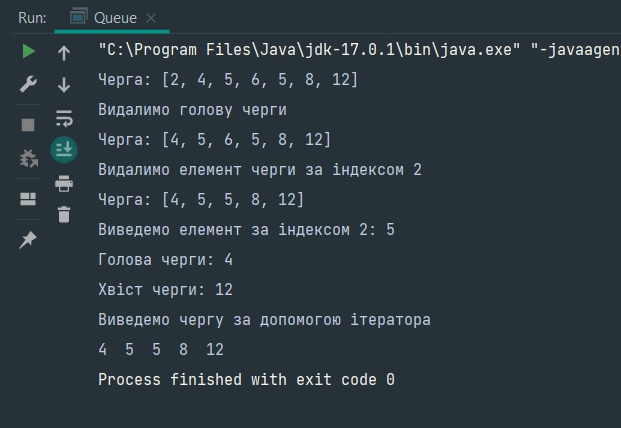


**Код програми**

**Queue**

*import* java.util.*Iterator*;  
*import* java.util.NoSuchElementException;  
  
*public class* Queue<E> *implements Iterable*<E> {  
  
  
 *public static void* main(String[] args) {  
  
 Queue<Integer> queue = *new* Queue<>();  
  
 queue.enqueue(2);  
 queue.enqueue(4);  
 queue.enqueue(5);  
 queue.enqueue(6);  
 queue.enqueue(5);  
 queue.enqueue(8);  
 queue.enqueue(12);  
  
 System.***out***.println("Черга: " + queue);  
 System.***out***.println("Видалимо голову черги");  
 queue.dequeue();  
 System.***out***.println("Черга: " + queue);  
 System.***out***.println("Видалимо елемент черги за індексом 2");  
 queue.remove(2);  
 System.***out***.println("Черга: " + queue);  
 System.***out***.println("Виведемо елемент за індексом 2: " + queue.get(2));  
  
  
 System.***out***.println("Голова черги: " + queue.head());  
 System.***out***.println("Хвіст черги: " + queue.tail());  
  
  
 System.***out***.println("Виведемо чергу за допомогою ітератора");  
 *for* (Integer i : queue) {  
 System.***out***.print(i + " ");  
 }  
  
 }  
  
  
 *private* Node<E> firstNode;  
 *private* Node<E> lastNode;  
 *private int* size;  
  
 *public* Queue() {*//конструктор* lastNode = *new* Node<>(*null*, *null*);  
 firstNode = *new* Node<>(*null*, *null*);  
 }  
  
 *public boolean* isEmpty() {*//метод для перевірки листа на наявність елементів  
 return* size == 0;  
 }  
  
 *public int* size() {*//метод для виведення розміру  
 return* size;  
 }  
  
 *public void* enqueue(E item) {*//метод для додавання елементу до списку* Node<E> oldlast = lastNode;  
 lastNode = *new* Node(item, *null*);  
 *if* (isEmpty()) firstNode = lastNode;  
 *else* oldlast.next = lastNode;  
 size++;  
  
 }  
  
 *public* E dequeue() {*//метод для видалення першого елементу списку  
 if* (isEmpty()) *throw new* NoSuchElementException("Queue is clear");  
 E item = firstNode.getElem();  
 firstNode = firstNode.next;  
 size--;  
 *if* (isEmpty()) lastNode = *null*;  
 *return* item;  
 }  
  
 *public void* remove(*int* index) {*//метод для видалення елементу за індексом  
 if* (index < 0 || index >= size) *throw new* NoSuchElementException("No element index");  
 *if* (index == 0) {  
 dequeue();  
  
 } *else* {  
  
 Node<E> temp = firstNode;  
 *for* (*int* i = 0; i < index - 1; i++) {  
 temp = temp.next;  
 }  
 Node<E> temp2 = temp.next;  
 temp.setNext(temp2.next);  
 temp2.setElem(*null*);  
 size--;  
  
 }  
  
 }  
  
  
 *public* E head() {*//повертаємо голову черги  
 if* (isEmpty()) *throw new* NoSuchElementException("Queue is clear");  
 *return* firstNode.getElem();  
 }  
  
 *public* E tail() {*//повертаємо хвіст черги  
 if* (isEmpty()) *throw new* NoSuchElementException("Queue is clear");  
 *return* lastNode.getElem();  
 }  
  
 *public* E get(*int* index) {*//метод для знаходження елемента списку за заданим індексом  
  
 if* (index < 0 || index >= size) *throw new* NoSuchElementException("No element index");  
 Node<E> temp = firstNode;  
 *for* (*int* i = 0; i < index; i++) {  
 temp = getNext(temp);  
 }  
 *return* temp.getElem();  
 }  
  
  
 @Override  
 *public* String toString() {*//Перевизначимо метод toString  
  
 if* (size == 0) *return* "Queue is clear";  
 StringBuilder sb = *new* StringBuilder();  
 sb.append("[");  
  
 Node<E> current = firstNode;  
  
 *while* (current != *null*) {  
 sb.append(current.getElem());  
 current = current.next;  
 *if* (current != *null*) sb.append(", ");  
 }  
  
 sb.append("]");  
 *return* sb.toString();  
 }  
  
 @Override  
 *public Iterator*<E> iterator() {*//визначимо ітератор  
 return new* Iterator<E>() {  
 *int* counter = 0;  
  
 @Override  
 *public boolean* hasNext() {  
 *return* counter < size;  
 }  
  
 @Override  
 *public* E next() {  
 *return* get(counter++);  
 }  
 };  
 }  
  
 *private* Node<E> getNext(Node<E> current) {  
 *return* current.getNext();  
 }  
  
 *private class* Node<E> {*//створимо клас Node  
  
 private* E elem;  
 *private* Node<E> next;  
  
 *private* Node(E elem, Node<E> next) {  
 *this*.elem = elem;  
 *this*.next = next;  
 }  
  
 *public* E getElem() {  
 *return* elem;  
 }  
  
 *public void* setElem(E elem) {  
 *this*.elem = elem;  
 }  
  
 *public* Node<E> getNext() {  
 *return* next;  
 }  
  
 *public void* setNext(Node<E> next) {  
 *this*.next = next;  
 }  
  
  
 }  
  
}

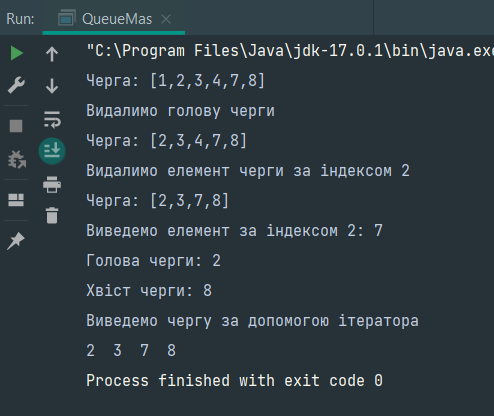
**Результат програми**

****

**QueueMas**

*import* java.util.*Iterator*;  
*import* java.util.NoSuchElementException;  
  
*public class* QueueMas<E> *implements Iterable*<E> {  
  
 *public static void* main(String[] args) {  
 QueueMas<Integer> queueMas = *new* QueueMas<>();  
  
 queueMas.enqueue(1);  
 queueMas.enqueue(2);  
 queueMas.enqueue(3);  
 queueMas.enqueue(4);  
 queueMas.enqueue(7);  
 queueMas.enqueue(8);  
 System.***out***.println("Черга: " + queueMas);  
 System.***out***.println("Видалимо голову черги");  
 queueMas.dequeue();  
 System.***out***.println("Черга: " + queueMas);  
 System.***out***.println("Видалимо елемент черги за індексом 2");  
 queueMas.remove(2);  
 System.***out***.println("Черга: " + queueMas);  
 System.***out***.println("Виведемо елемент за індексом 2: " + queueMas.get(2));  
  
  
 System.***out***.println("Голова черги: " + queueMas.head());  
 System.***out***.println("Хвіст черги: " + queueMas.tail());  
  
  
 System.***out***.println("Виведемо чергу за допомогою ітератора");  
 *for* (Integer i : queueMas) {  
 System.***out***.print(i + " ");  
 }  
  
 }  
  
  
 *private* E[] data;  
 *private final int* size = 100;  
 *private int* currentSize;  
 *private int* counter = 0;  
  
 *public* QueueMas() {*//конструктор* data = (E[]) *new* Object[size];  
 }  
  
 *public boolean* isEmpty() {  
 *return* currentSize == 0;  
 }  
  
 *public int* size() {  
 *return* currentSize;  
 }  
  
 *public void* enqueue(E elem) {*//метод для додавання елементу до списку  
  
 if* (currentSize == 100) *throw new* NoSuchElementException("No size");  
  
 data[counter] = elem;  
 counter++;  
 currentSize++;  
 }  
  
 *public void* dequeue() {*//метод для видалення першого елементу списку* E[] newData = (E[]) *new* Object[size];  
 *for* (*int* i = 1; i < currentSize; i++) {  
 newData[i - 1] = data[i];  
 }  
 currentSize--;  
 data = newData;  
 *this*.counter--;  
 }  
  
  
 *public void* remove(*int* index) {*//метод для видалення елементу за індексом* E[] newData = (E[]) *new* Object[size];  
 *for* (*int* i = 0; i < index; i++) {  
 newData[i] = data[i];  
 }  
 *for* (*int* i = index; i < currentSize; i++) {  
 newData[i] = data[i + 1];  
 }  
 currentSize--;  
 data = newData;  
 *this*.counter--;  
  
 }  
  
 *public* E head() {*//повертаємо голову черги  
 if* (isEmpty()) *throw new* NoSuchElementException("Queue is clear");  
 *return* data[0];  
 }  
  
 *public* E tail() {*//повертаємо хвіст черги  
 if* (isEmpty()) *throw new* NoSuchElementException("Queue is clear");  
 *return* data[currentSize - 1];  
 }  
  
 *public* E get(*int* index) {*//метод для знаходження елемента списку за заданим індексом  
  
 if* (index < 0 || index >= currentSize) *throw new* NoSuchElementException("No element index");  
 *return* data[index];  
  
 }  
  
 @Override  
 *public* String toString() {*//Перевизначимо метод toString  
  
 if* (currentSize == 0) *return* "Queue is clear";  
 StringBuilder sb = *new* StringBuilder();  
 sb.append("[");  
  
  
 *for* (*int* i = 0; i < currentSize; i++) {  
  
 sb.append(data[i]);  
  
 *if* (i != currentSize - 1) {  
 sb.append(",");  
 }  
 }  
 sb.append("]");  
 *return* sb.toString();  
 }  
  
  
 @Override  
 *public Iterator*<E> iterator() {*//визначимо ітератор  
 return new* Iterator() {  
 *int* counter = 0;  
  
 @Override  
 *public boolean* hasNext() {  
 *return* counter < currentSize;  
 }  
  
 @Override  
 *public* E next() {  
 *return* data[counter++];  
 }  
 };  
 }  
}

**Результат програми**

****