```java
/*

PUC Minas - Ciencia da Computacao     Nome: Stack

Autor: Axell Brendow Batista Moreira  Matricula: 631822

Versao:  1.0                          Data: 19/09/2018

*/

class Stack
{
    Cell _top;

    public Stack()
    {
        _top = null; // redundante em Java
    }

    private Cell getTop()
    {
        return _top;
    }

    private void setTop(Cell top)
    {
        _top = top;
    }

    public int sumElements()
    {
        int sum = 0;
        Cell currentCell = getTop();

        while (currentCell != null)
        {
            sum += (int) currentCell.getElement();

            currentCell = currentCell.getNext();
        }

        return sum;
    }

    public int sumElementsRecursively(Cell currentCell)
    {
        int sum = 0;

        if (currentCell != null)
        {
            sum += (int) currentCell.getElement();

            sum += sumElementsRecursively(currentCell.getNext());
        }

        return sum;
    }

    public int getGreatestElement()
    {
        int greatestElement = Integer.MIN_VALUE;
        int currentElement;
        Cell currentCell = getTop();

        while (currentCell != null)
        {
            currentElement = (int) currentCell.getElement();

            if (currentElement > greatestElement)
            {
                greatestElement = currentElement;
            }

            currentCell = currentCell.getNext();
        }

        return greatestElement;
    }

    public int getGreatestBetweenPreviousAndCurrent(Cell currentCell, int previousGreatestElement)
    {
        int greatestElement = previousGreatestElement;

        if (currentCell != null)
        {
            int currentElement = (int) currentCell.getElement();

            if (currentElement > greatestElement)
            {
                greatestElement = currentElement;
            }

            greatestElement = getGreatestBetweenPreviousAndCurrent(currentCell.getNext(), greatestElement);
        }

        return greatestElement;
    }

    public int getGreatestElementRecursively()
    {
        return getGreatestBetweenPreviousAndCurrent(getTop(), Integer.MIN_VALUE);
    }

    public void printGoingToBase(Cell currentCell)
    {
        if (currentCell != null)
        {
            System.out.println(currentCell.getElement().toString());

            printGoingToBase(currentCell.getNext());
        }
    }

    public void printFromTopToBase()
    {
        printGoingToBase(getTop());
    }

    public void printFromBaseToCell(Cell currentCell)
    {
        if (currentCell != null)
        {
            printFromBaseToCell(currentCell.getNext());

            System.out.println(currentCell.getElement().toString());
        }
    }

    public void printFromBaseToTop()
    {
        printFromBaseToCell(getTop());
    }

    public int getNumberOfElements()
    {
        int numberOfElements = 0;
        Cell currentCell = getTop();

        while (currentCell != null)
        {
            numberOfElements++;
            currentCell = currentCell.getNext();
        }
```

```java

            return numberOfElements;
        }

        public Cell getCellOnIndex(int index)
        {
            int currentIndex = 0;
            Cell currentCell = getTop();

            while (currentIndex < index)
            {
                currentIndex++;
                currentCell = currentCell.getNext();
            }

            return currentCell;
        }

        public int getCellIndex(Cell cellToFind, int numberOfElements)
        {
            int cellIndex = -1;
            Cell currentCell = getTop();

            for (int i = 0; i < numberOfElements; i++)
            {
                if (currentCell == cellToFind)
                {
                    cellIndex = i;
                    i = numberOfElements;
                }

                else
                {
                    currentCell = currentCell.getNext();
                }
            }

            return cellIndex;
        }

        public void printFromBaseToCellIteratively(Cell cell)
        {
            int numberOfElements = getNumberOfElements();
            int cellIndex = getCellIndex(cell, numberOfElements);

            for (int i = numberOfElements - 1; i >= cellIndex; i--)
            {
                System.out.println(getCellOnIndex(i).getElement().toString());
            }
        }

        public void printFromBaseToTopIteratively()
        {
            printFromBaseToCellIteratively(getTop());
        }
    }
```