

Programación II

Universidad Mesoamericana, Sede Quetzaltenango
Ingeniería en Sistemas, Informática y Ciencias de la
Computación

Ejercicios GUI



Castillo Tovar, Derek Andre - 202208060

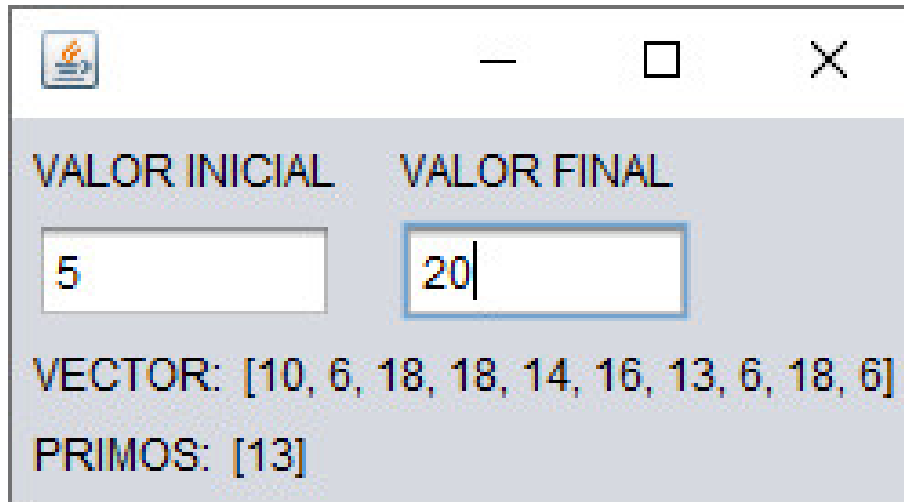
Contents

1	Números Primos	2
2	Números Abundantes, Exactos y Deficientes	3
3	Números Pares e Impares	5
4	Suma Diagonal-Principal - 5×5	7
5	Números Pares e Impares - 5×5	9

1 Números Primos

Llenar un vector de tipo integer de 10 números aleatorios entre un rango que el usuario define y mostrar que elementos son primos

Runtime



Codigo Implementado

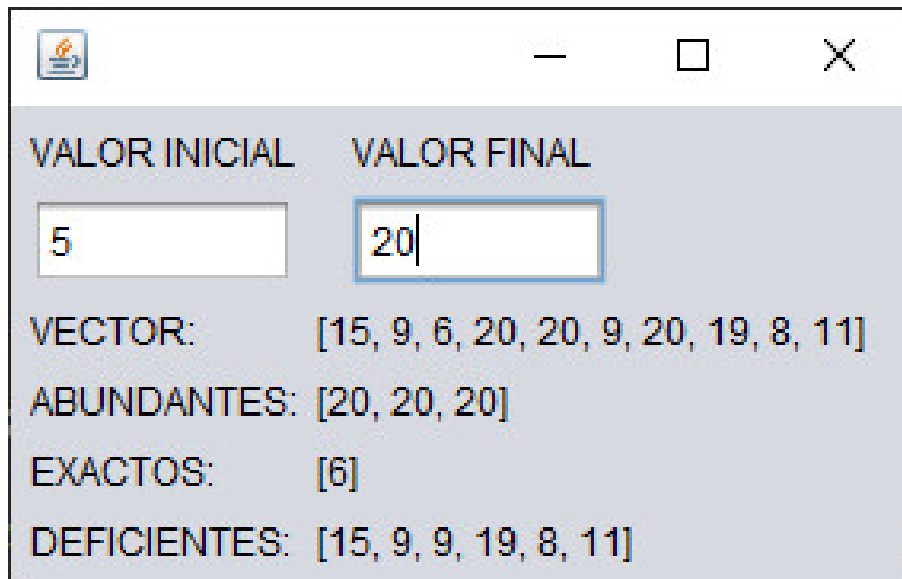
```
public static int[] getPrimes(int[] v){
    List<Integer> primes = new ArrayList<Integer>();
    for(int i : v){
        boolean prime=true;
        for(int j=2; j<i; j++){
            if(i%j==0){ prime=false; break; }
        }
        if(prime) primes.add(i);
    }
    int[] array = new int[primes.size()];
    for(int i=0; i<primes.size(); i++) array[i] = primes.get(i);
    return array;
}

public static void GlobalKeyRelease(KeyEvent e){
    int init=Integer.parseInt(jTFInit.getText()), fin=Integer.parseInt(jTFFin.getText());
    if(init==fin) {e.consume(); return;}
    jLVectOut.setText("");
    Random r = new Random();
    for(int i=0; i<10; i++){
        v[i]= init + r.nextInt(fin - init + 1);
    }
    jLVectOut.setText(Arrays.toString(v));
    int[] primes = getPrimes(v);
    jLPrimesOut.setText(Arrays.toString(primes));
}
```

2 Números Abundantes, Exactos y Deficientes

Llenar un vector de tipo integer de 10 números aleatorios entre un rango que el usuario define y mostrar que elementos son abundantes, exactos o deficientes

Runtime



The screenshot shows a Java application window with a title bar containing a logo, a minus sign, a maximize button, and a close button. The window has a light blue background and contains the following text:

VALOR INICIAL	VALOR FINAL
<input type="text" value="5"/>	<input type="text" value="20"/>
VECTOR:	[15, 9, 6, 20, 20, 9, 20, 19, 8, 11]
ABUNDANTES:	[20, 20, 20]
EXACTOS:	[6]
DEFICIENTES:	[15, 9, 9, 19, 8, 11]

Codigo Implementado

```

public static int[] getFactors(int v){
    List<Integer> factors = new ArrayList<Integer>();
    for(int i=1; i<v; i++) if(v%i==0) factors.add(i);
    int[] rtn = ListToArray(factors);
    return rtn;
}

public static void GlobalKeyRelease(KeyEvent e){
    int init=Integer.parseInt(jTextField1.getText()), fin=Integer.parseInt(jTextField2.getText());
    if(init==fin) {e.consume(); return;}
    lVector.setText("");
    Random r = new Random();
    for(int i=0; i<10; i++){
        v[i]= init + r.nextInt(fin - init + 1);
    }
    lVector.setText(Arrays.toString(v));
    List<Integer> Abundants = new ArrayList<Integer>(), Exacts = new ArrayList<Integer>(), Deficients = new ArrayList<Integer>();

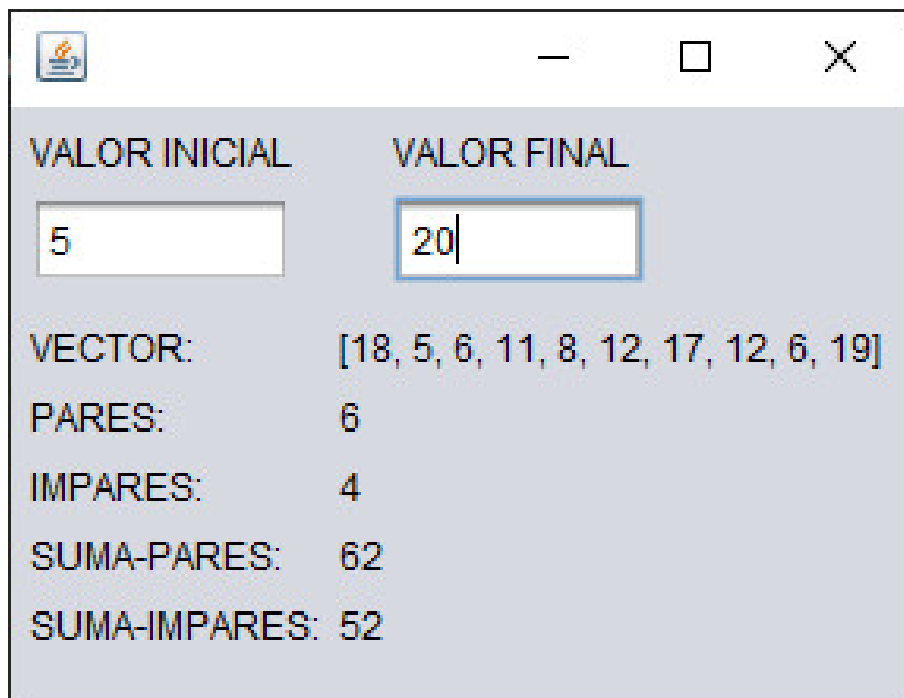
    for(int element : v) {
        int[] factors = getFactors(element);
        int sum = 0;
        for(int i : factors) sum+=i;
        if(sum>element) Abundants.add(element);
        else if(sum==element) Exacts.add(element);
        else if(sum<element) Deficients.add(element);
    }
    int[] Abu = ListToArray(Abundants), Exa = ListToArray(Exacts), Def = ListToArray(Deficients);
    lAbundant.setText(Arrays.toString(Abu));
    lExact.setText(Arrays.toString(Exa));
    lDeficient.setText(Arrays.toString(Def));
}

```

3 Números Pares e Impares

Llenar un vector de tipo integer de 10 números aleatorios entre un rango que el usuario define y mostrar la cantidad de elementos que son pares e impares más la suma de los elementos pares e impares

Runtime



The screenshot shows a Java application window with a title bar containing a Java logo, a minus sign, a maximize button, and a close button. The window has a light blue background and contains the following text and input fields:

VALOR INICIAL	VALOR FINAL
5	20
VECTOR:	[18, 5, 6, 11, 8, 12, 17, 12, 6, 19]
PARES:	6
IMPARES:	4
SUMA-PARES:	62
SUMA-IMPARES:	52

Codigo Implementado

```

public static int[] getFactors(int v){
    List<Integer> factors = new ArrayList<Integer>();
    for(int i=1; i<v; i++) if(v%i==0) factors.add(i);
    int[] rtn = ListToArray(factors);
    return rtn;
}

public static void GlobalKeyRelease(KeyEvent e){
    int init=Integer.parseInt(jTextField1.getText()), fin=Integer.parseInt(jTextField2.getText());
    if(init>fin) {e.consume(); return;}
    lVector.setText("");
    Random r = new Random();
    for(int i=0; i<10; i++)
        v[i]= init + r.nextInt(fin - init + 1);
    lVector.setText(Arrays.toString(v));
    List<Integer> Abundants = new ArrayList<Integer>(), Exacts = new ArrayList<Integer>(), Deficients = new ArrayList<Integer>();

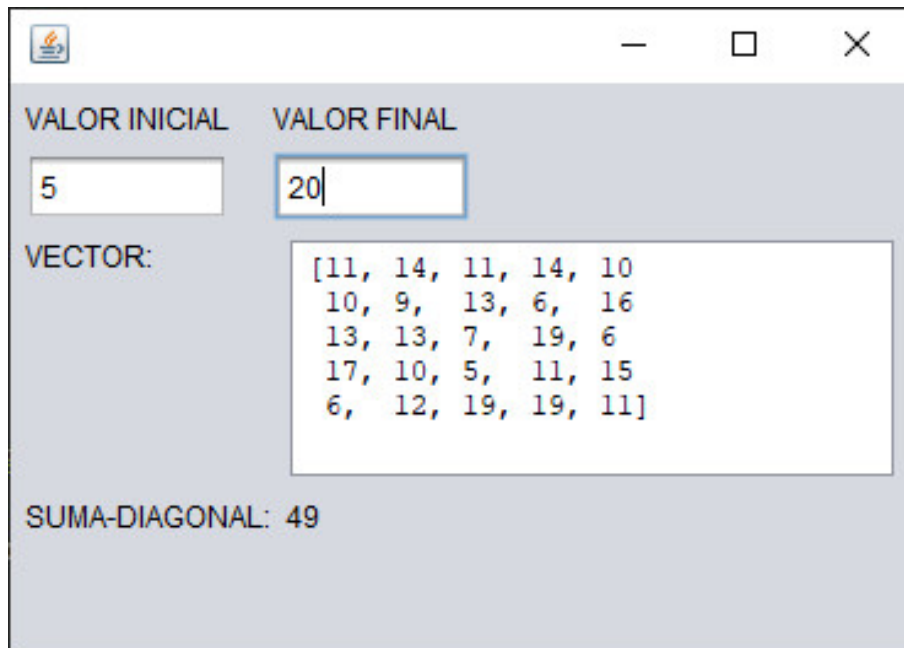
    for(int element : v) {
        int[] factors = getFactors(element);
        int sum = 0;
        for(int i : factors) sum+=i;
        if(sum>element) Abundants.add(element);
        else if(sum==element) Exacts.add(element);
        else if(sum<element) Deficients.add(element);
    }
    int[] Abu = ListToArray(Abundants), Exa = ListToArray(Exacts), Def = ListToArray(Deficients);
    lAbundant.setText(Arrays.toString(Abu));
    lExact.setText(Arrays.toString(Exa));
    lDeficient.setText(Arrays.toString(Def));
}

```

4 Suma Diagonal-Principal - 5×5

Llenar una matriz 5×5 de tipo integer de números aleatorios entre un rango que el usuario define y mostrar la suma de la diagonal principal

Runtime



VALOR INICIAL VALOR FINAL

5 20

VECTOR:

```
[11, 14, 11, 14, 10  
10, 9, 13, 6, 16  
13, 13, 7, 19, 6  
17, 10, 5, 11, 15  
6, 12, 19, 19, 11]
```

SUMA-DIAGONAL: 49

Codigo Implementado


```

public String ArrayToString(int[][] a){
    int checker=Integer.MIN_VALUE;
    for(int i=0; i<5; i++){
        if(checker < a[i/5][i%5])
            checker=a[i/5][i%5];
    }
    String str="[";
    for(int i=0; i<5; i++){
        for(int j=0; j<5; j++){
            char[] cv = new char[Integer.toString(checker).length() - Integer.toString(a[i][j]).length()];
            Arrays.fill(cv, ' ');
            str += a[i][j] + (j+1<5 ? ", " : "") + new String(cv);
        }
        str += i+1<5 ? "\n " : "";
    }
    str+="]";
    return str;
}

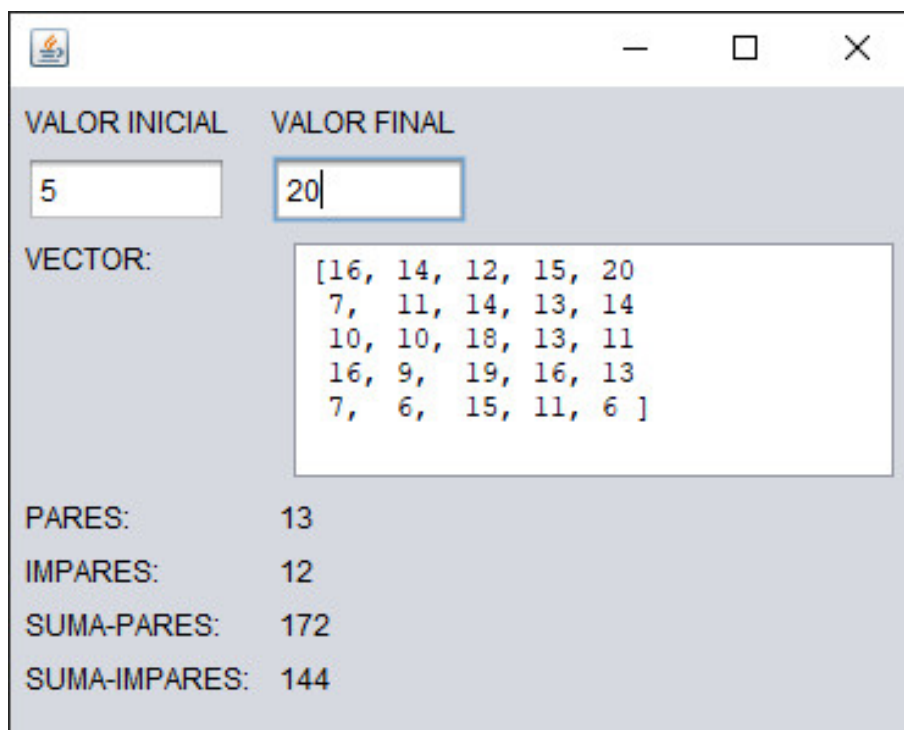
public void GlobalKeyRelease(KeyEvent e){
    int init=Integer.parseInt(jTextField1.getText()), fin=Integer.parseInt(jTextField2.getText()), sum=0;
    if(init>=fin) {e.consume(); return;}
    tpVector.setText("");
    Random r = new Random();
    for(int i=0; i<25; i++){
        v[i/5][i%5]= init + r.nextInt(fin - init + 1);
    }
    tpVector.setText(ArrayToString(v));
    for(int i=0; i<5; i++) sum+=v[i][i];
    lSuma.setText(Integer.toString(sum));
}

```

5 Números Pares e Impares - 5×5

Llenar una matriz 5×5 de tipo integer de números aleatorios entre un rango que el usuario define y mostrar la cantidad de elementos que son pares e impares más la suma de los elementos pares e impares

Runtime



The screenshot shows a Java Swing window with a light gray background. At the top, there are two labels: 'VALOR INICIAL' and 'VALOR FINAL'. Below 'VALOR INICIAL' is a text field containing the number '5'. Below 'VALOR FINAL' is a text field containing the number '20'. Below these fields is a label 'VECTOR:' followed by a white rectangular box containing a 5x5 matrix of numbers. Below the matrix box, there are four lines of text: 'PARES: 13', 'IMPARES: 12', 'SUMA-PARES: 172', and 'SUMA-IMPARES: 144'.

VALOR INICIAL	VALOR FINAL
5	20

VECTOR:

```
[16, 14, 12, 15, 20  
7, 11, 14, 13, 14  
10, 10, 18, 13, 11  
16, 9, 19, 16, 13  
7, 6, 15, 11, 6 ]
```

PARES: 13
IMPARES: 12
SUMA-PARES: 172
SUMA-IMPARES: 144

Codigo Implementado

```

public String ArrayToString(int[][] a){
    int checker=Integer.MIN_VALUE;
    for(int i=0; i<5; i++){
        if(checker < a[i/5][i%5])
            checker=a[i/5][i%5];
        String str="[";
        for(int i=0; i<5; i++){
            for(int j=0; j<5; j++){
                char[] cv = new char[Integer.toString(checker).length() - Integer.toString(a[i][j]).length()];
                Arrays.fill(cv, ' ');
                str += a[i][j] + (j+1<5 ? ", " : "") + new String(cv);
            }
            str += i+1<5 ? "\n " : "";
        }
        str+="]";
        return str;
    }
}

public void GlobalKeyRelease(KeyEvent e){
    int init=Integer.parseInt(jTextField1.getText()), fin=Integer.parseInt(jTextField2.getText());
    if(init>=fin) {e.consume(); return;}
    tpVector.setText("");
    Random r = new Random();
    for(int i=0; i<25; i++){
        v[i/5][i%5] = init + r.nextInt(fin - init + 1);
        tpVector.setText(ArrayToString(v));
        int psum=0, ipsum=0, pcount=0, ipcount=0;
        for(int[] row : v)
            for(int el : row){
                if(el%2==0){
                    pcount++;
                    psum+=el;
                }else{
                    ipcount++;
                    ipsum+=el;
                }
            }
    }
    lPares.setText(Integer.toString(pcount));
    lImpares.setText(Integer.toString(ipcount));
    lSPares.setText(Integer.toString(psum));
    lSImpares.setText(Integer.toString(ipsum));
}

```