Emri: _____

ID (Nr. dosjes): _____

Drejtimi: _____

Data: _____

Struktura e të dhënave

Provimi përfundimtar, Forma: A

1. Është dhënë klasa

```
import javax.swing.*;
/** Numëron votat për kandidatët elektoralë.
 *  input: një varg votash, i terminuar nga -1
 *  output: lista e rezultateve të votave për kandidatë */
public class VoteCount
{ public static void main(String[] args)
  { int numCandidates = 4;
    int[] votes = new int[numCandidates];
    boolean processing = true;
    while ( processing )
    { int v = new Integer(JOptionPane.showInputDialog
                ("Votoni për (0,1,2,3):")).intValue();
      if ( v == -1 )
      { processing = false; }
      else if ( v >= 0 && v < numCandidates )
      { votes[v] = votes[v] + 1; }
      else { JOptionPane.showMessageDialog(null,
                "Gabim në votim: " + v);
           }
    }
    for ( int i = 0; i < numCandidates; i = i + 1 )
    { System.out.println("Kandidati " + i + " ka "
          + votes[i] + " vota"); }
  }
}
```

   Modifikoni `class VoteCount` ashtu që aplikacioni të afishojë numrin e votave për secilin kandidat, si dhe kandidatin fitues. (Në qoftë se ka më tepër kandidatë me numër maksimal votash, atëherë të afishohet mesazhi për mungesë fituesi.)

2. Është dhënë metoda

```
public String reverse(String s)
{ String answer = "";
  int last = s.length() - 1;
  for (int i = last; i >= 0; i = i - 1)
  { answer = answer + s.charAt(i); }
  return answer;
}
```

   Çfarë kthen invokimi `reverse("hehs e kun a")`?

3. Krijoni një varg prej 50 numrash `double` të tillë që vlera e `d[i]` të jetë $\sqrt{i}$.

Për pyetjet 4–5 shqyrtoni metodën vijuese.

```
public int[] primes(int n)
{ int [] answer = new int[n];
  answer[0] = 2;
  answer[1] = 3;
  int count = 2;
  int next = 5;
  while ( count < n )
  { int limit = (int)(Math.sqrt(next));
    boolean found = false;
    int i = 1;
    while ( !found && answer[i] <= limit)
    { if ( next % answer[i] == 0 )
      { found = true; }
      else { i = i + 1; }
    }
    if ( !found )
    { answer[count] = next;
      count = count + 1;
    }
    next = next + 2;
  }
  return answer;
}
```

4. Çfarë kthen invokimi `primes(5)`?

5. Çfarë kthehet si rezultat i invokimit `primes(n)` për argumentin `n` numër të plotë më të madh se 1? Shkruani invariantat e të dy iterimeve në metodën. Shpjegoni algoritmin e metodës.

6. Shkruani një klasë e cila gjeneron një elipsë me diametër horizontal 300 piksel dhe diametër vertikal 200 piksel, brenda së cilës ndodhet një elipsë tjetër e madhësisë 0.8 madhësi të së parës, brenda së cilës ndodhet një elipsë tjetër e madhësisë 0.8 madhësi të së dytës, ..., derisa elipsat të mos tkurren në madhësinë 0.

7. Çfarë do të afishojë urdhëri `while` vijues?

```
int number = 2000 ;
int i = 10 ;
while ( i <= 50  )
{
  if ( number % i == 0)
  { System.out.println(i); }
  i = i + 1;
}
```

Çelësi i provimit A

1. Është dhënë klasa

```
import javax.swing.*;
/** Numëron votat për kandidatët elektoralë.
 *  input: një varg votash, i terminuar nga -1
 *  output: lista e rezultateve të votave për kandidatë */
public class VoteCount
{ public static void main(String[] args)
  { int numCandidates = 4;
    int[] votes = new int[numCandidates];
    boolean processing = true;
    while ( processing )
    { int v = new Integer(JOptionPane.showInputDialog
                ("Votoni për (0,1,2,3):")).intValue();
      if ( v == -1 )
      { processing = false; }
      else if ( v >= 0 && v < numCandidates )
      { votes[v] = votes[v] + 1; }
      else { JOptionPane.showMessageDialog(null,
                "Gabim në votim: " + v);
           }
    }
    for ( int i = 0; i < numCandidates; i = i + 1 )
    { System.out.println("Kandidati " + i + " ka "
          + votes[i] + " vota"); }
  }
}
```

Modifikoni `class VoteCount` ashtu që aplikacioni të afishojë numrin e votave për secilin kandidat, si dhe kandidatin fitues. (Në qoftë se ka më tepër kandidatë me numër maksimal votash, atëherë të afishohet mesazhi për mungesë fituesi.)

Përgjegjja:

```
import javax.swing.*;
/** Numëron votat për kandidatët elektoralë.
 *  input: një varg votash, i terminuar nga -1
 *  output: lista e rezultateve të votave për kandidatë */
public class VoteCount
{ public static void main(String[] args)
  { // ... sikur më parë
    if ( numCandidates <= 0 )
    { // ... sikur më parë
    }
    else
    { // ... sikur më parë
      int maxVotesIndex = 0;
      boolean tied = false;
      for ( int i = 1; i < numCandidates; i++ )
      { if ( votes[i] > votes[maxVotesIndex] )
```

```
              { maxVotesIndex = i;
                tied = false;
              }
              else if ( votes[i] == votes[maxVotesIndex] )
              { tied = true; }
            }
            if ( tied )
            { System.out.println("Nuk ka fitues"); }
            else
            { System.out.println("Fitues është kandidati "
                + names[maxVotesIndex]); }
          }
        }
      }
```

2. Është dhënë metoda

   ```
   public String reverse(String s)
   { String answer = "";
     int last = s.length() - 1;
     for (int i = last; i >= 0; i = i - 1)
     { answer = answer + s.charAt(i); }
     return answer;
   }
   ```

   Çfarë kthen invokimi `reverse("hehs e kun a")`?

   Përgjegjja: `"a nuk e sheh"`

3. Krijoni një varg prej 50 numrash `double` të tillë që vlera e `d[i]` të jetë $\sqrt{i}$.

   Përgjegjja:

Për pyetjet 4–5 shqyrtoni metodën vijuese.

```
public int[] primes(int n)
{ int [] answer = new int[n];
  answer[0] = 2;
  answer[1] = 3;
  int count = 2;
  int next = 5;
  while ( count < n )
  { int limit = (int)(Math.sqrt(next));
    boolean found = false;
    int i = 1;
    while ( !found && answer[i] <= limit)
    { if ( next % answer[i] == 0 )
      { found = true; }
      else { i = i + 1; }
    }
    if ( !found )
    { answer[count] = next;
      count = count + 1;
    }
    next = next + 2;
  }
```

```
  return answer;
}
```

4. Çfarë kthen invokimi `primes(5)`?

   Përgjegjja: Vargun me termat `2,3,5,7,11`

5. Çfarë kthehet si rezultat i invokimit `primes(n)` për argumentin `n` numër të plotë më të madh se 1? Shkruani invariantat e të dy iterimeve në metodën. Shpjegoni algoritmin e metodës.

   Përgjegjja:

```
/** Kthen vargun e numrave të thjeshtë të gjatësisë sa argumenti i dhënë
 *  @param n - numri i dhënë natyror
 *  @return vargu i n numrave të parë të thjeshtë */
public int[] primes(int n)
{ int [] answer = new int[n];
  answer[0] = 2;
  answer[1] = 3;
  int count = 2;
  int next = 5;
  while ( count < n )
  // Vlerat e a[0],a[1],...,a[count-1] janë me rradhë count-1
  // numrat e parë të thjeshtë
  { int limit = (int)(Math.sqrt(next));
    boolean found = false;
    int i = 1;
    while ( !found && answer[i] <= limit)
    // Përderisa found == false,
    // asnjëri nga numrat e thjeshtë a[0],a[1],...,a[i-1]
    // nuk është pjesëtues i next
    { if ( next % answer[i] == 0 )
      { found = true; }
      else { i = i + 1; }
    }
    if ( !found )
    { answer[count] = next;
      count = count + 1;
    }
    next = next + 2;
  }
  return answer;
}
```

   Algoritmi kërkon për vlerat teke duke filluar nga 5 të `next` se a kanë për pjesëtues ndonjërin nga numrat e thjeshtë (më të mëdhenjë se 2 e të cilët nuk e tejkalojnë $\sqrt{\texttt{next}}$) të gjetur më parë. Në qoftë se gjendet një pjesëtues, atëherë kalohet në vlerën vijuese të `next`; në të kundërtën vlera `next` ruhet si vlerë vijuese e thjeshtë në elementin vijues të vargut `answer`.

6. Shkruani një klasë e cila gjeneron një elipsë me diametër horizontal 300 piksel dhe diametër vertikal 200 piksel, brenda së cilës ndodhet një elipsë tjetër e madhësisë 0.8 madhësi të së parës, brenda së cilës ndodhet një elipsë tjetër e madhësisë 0.8 madhësi të së dytës, ..., derisa elipsat të mos tkurren në madhësinë 0.

   Përgjegjja:

```
import javax.swing.*;
import java.awt.*;
/** Afishon figurën rekursive të elipsave */
public class RecursiveOvalWriter extends JPanel
{ private double scale = 0.8;
  private int width = 600;
  private int height = 400;
  /** Krijon dritaren */
  public RecursiveOvalWriter()
  { JFrame frame = new JFrame();
    frame.getContentPane().add(this);
    frame.setTitle("Vizatuesi rekursiv i elipsave");
    frame.setSize(width, height);
    frame.setBackground(Color.white);
    frame.setVisible(true);
  }
  /** Vizaton figurën rekursive
   * @param g - penda grafike */
  public void paintComponent(Graphics g)
  { paintOval(300, g); }
  /**
   * @param ovalWidth - gjerësia e elipsës
   * @param g - penda grafike */
  private void paintOval(int ovalWidth, Graphics g)
  { int backgroundSize = (int)(ovalWidth * scale);
    int backgroundHeight = (int)(2 * backgroundSize / 3);
    if ( backgroundHeight > 0 )
    { paintOval(backgroundSize, g);
    }
    int ovalHeight = (int)(2 * ovalWidth / 3);
    int top = (height - ovalHeight) / 2;
    g.setColor(Color.black);
    g.drawOval(0, top, ovalWidth, ovalHeight);
  }

  public static void main(String[] args)
  { new RecursiveOvalWriter(); }
}
```

7. Çfarë do të afishojë urdhëri `while` vijues?

```
int number = 2000 ;
int i = 10 ;
while ( i <= 50  )
{
  if ( number % i == 0)
  { System.out.println(i); }
  i = i + 1;
}
```

Përgjegjja: Të gjithë pjesëtuesit e numrit 2000 të cilët ndodhen ndërmjet numrave 10 dhe 50 .

Emri: _____

ID (Nr. dosjes): _____

Drejtimi: _____

Data: _____

Struktura e të dhënave

Provimi përfundimtar, Forma: B

1. Është dhënë metoda

```
public String reverse(String s)
{ String answer = "";
  int last = s.length() - 1;
  for (int i = last; i >= 0; i = i - 1)
  { answer = answer + s.charAt(i); }
  return answer;
}
```

   Çfarë kthen invokimi `reverse("ah ah ah")`?

2. Krijoni një varg prej 100 numrash `double` të tillë që vlera e `d[i]` të jetë $\sqrt{i}$.

3. Është dhënë klasa

```
import javax.swing.*;
/** Numëron votat për kandidatët elektoralë.
 *  input: një varg votash, i terminuar nga -1
 *  output: lista e rezultateve të votave për kandidatë */
public class VoteCount
{ public static void main(String[] args)
  { int numCandidates = 4;
    int[] votes = new int[numCandidates];
    boolean processing = true;
    while ( processing )
    { int v = new Integer(JOptionPane.showInputDialog
                ("Votoni për (0,1,2,3):")).intValue();
      if ( v == -1 )
      { processing = false; }
      else if ( v >= 0 && v < numCandidates )
      { votes[v] = votes[v] + 1; }
      else { JOptionPane.showMessageDialog(null,
                "Gabim në votim: " + v);
           }
    }
    for ( int i = 0; i < numCandidates; i = i + 1 )
    { System.out.println("Kandidati " + i + " ka "
          + votes[i] + " vota"); }
  }
}
```

   Modifikoni `class VoteCount` ashtu që aplikacioni të afishojë numrin e votave për secilin kandidat, si dhe kandidatin fitues. (Në qoftë se ka më tepër kandidatë me numër maksimal votash, atëherë të afishohet mesazhi për mungesë fituesi.)

4. Shkruani një klasë e cila gjeneron një elipsë me diametër horizontal 300 piksel dhe diametër vertikal 200 piksel, brenda së cilës ndodhet një elipsë tjetër e madhësisë 0.8 madhësi të së parës, brenda së cilës ndodhet një elipsë tjetër e madhësisë 0.8 madhësi të së dytës, ..., derisa elipsat të mos tkurren në madhësinë 0.

Për pyetjet 5–6 shqyrtoni metodën vijuese.

```
public int[] primes(int n)
{ int [] answer = new int[n];
  answer[0] = 2;
  answer[1] = 3;
  int count = 2;
  int next = 5;
  while ( count < n )
  { int limit = (int)(Math.sqrt(next));
    boolean found = false;
    int i = 1;
    while ( !found && answer[i] <= limit)
    { if ( next % answer[i] == 0 )
      { found = true; }
      else { i = i + 1; }
    }
    if ( !found )
    { answer[count] = next;
      count = count + 1;
    }
    next = next + 2;
  }
  return answer;
}
```

5. Çfarë kthen invokimi `primes(4)`?

6. Çfarë kthehet si rezultat i invokimit `primes(n)` për argumentin `n` numër të plotë më të madh se 1? Shkruani invariantat e të dy iterimeve në metodën. Shpjegoni algoritmin e metodës.

7. Çfarë do të afishojë urdhëri `while` vijues?

```
int number = 3000 ;
int i = 50 ;
while ( i <= 100  )
{
  if ( number % i == 0)
  { System.out.println(i); }
  i = i + 1;
}
```

Çelësi i provimit B

1. Është dhënë metoda

```
public String reverse(String s)
{ String answer = "";
  int last = s.length() - 1;
  for (int i = last; i >= 0; i = i - 1)
  { answer = answer + s.charAt(i); }
  return answer;
}
```

Çfarë kthen invokimi `reverse("ah ah ah")`?

Përgjegjja: `"ha ha ha"`

2. Krijoni një varg prej 100 numrash `double` të tillë që vlera e `d[i]` të jetë $\sqrt{i}$.

Përgjegjja:

3. Është dhënë klasa

```
import javax.swing.*;
/** Numëron votat për kandidatët elektoralë.
 *  input: një varg votash, i terminuar nga -1
 *  output: lista e rezultateve të votave për kandidatë */
public class VoteCount
{ public static void main(String[] args)
  { int numCandidates = 4;
    int[] votes = new int[numCandidates];
    boolean processing = true;
    while ( processing )
    { int v = new Integer(JOptionPane.showInputDialog
                ("Votoni për (0,1,2,3):")).intValue();
      if ( v == -1 )
      { processing = false; }
      else if ( v >= 0 && v < numCandidates )
      { votes[v] = votes[v] + 1; }
      else { JOptionPane.showMessageDialog(null,
                "Gabim në votim: " + v);
           }
    }
    for ( int i = 0; i < numCandidates; i = i + 1 )
    { System.out.println("Kandidati " + i + " ka "
          + votes[i] + " vota"); }
  }
}
```

Modifikoni `class VoteCount` ashtu që aplikacioni të afishojë numrin e votave për secilin kandidat, si dhe kandidatin fitues. (Në qoftë se ka më tepër kandidatë me numër maksimal votash, atëherë të afishohet mesazhi për mungesë fituesi.)

Përgjegjja:

```
import javax.swing.*;
/** Numëron votat për kandidatët elektoralë.
 *  input: një varg votash, i terminuar nga -1
 *  output: lista e rezultateve të votave për kandidatë */
public class VoteCount
{ public static void main(String[] args)
  { // ... sikur më parë
    if ( numCandidates <= 0 )
    { // ... sikur më parë
    }
    else
    { // ... sikur më parë
      int maxVotesIndex = 0;
      boolean tied = false;
      for ( int i = 1; i < numCandidates; i++ )
      { if ( votes[i] > votes[maxVotesIndex] )
        { maxVotesIndex = i;
          tied = false;
        }
        else if ( votes[i] == votes[maxVotesIndex] )
        { tied = true; }
      }
      if ( tied )
      { System.out.println("Nuk ka fitues"); }
      else
      { System.out.println("Fitues është kandidati "
          + names[maxVotesIndex]); }
    }
  }
}
```

4. Shkruani një klasë e cila gjeneron një elipsë me diametër horizontal 300 piksel dhe diametër vertikal 200 piksel, brenda së cilës ndodhet një elipsë tjetër e madhësisë 0.8 madhësi të së parës, brenda së cilës ndodhet një elipsë tjetër e madhësisë 0.8 madhësi të së dytës, ..., derisa elipsat të mos tkurren në madhësinë 0.

   Përgjegjja:

```
import javax.swing.*;
import java.awt.*;
/** Afishon figurën rekursive të elipsave */
public class RecursiveOvalWriter extends JPanel
{ private double scale = 0.8;
  private int width = 600;
  private int height = 400;
  /** Krijon dritaren */
  public RecursiveOvalWriter()
  { JFrame frame = new JFrame();
    frame.getContentPane().add(this);
    frame.setTitle("Vizatuesi rekursiv i elipsave");
    frame.setSize(width, height);
    frame.setBackground(Color.white);
    frame.setVisible(true);
  }
  /** Vizaton figurën rekursive
```

```
     * @param g - penda grafike */
    public void paintComponent(Graphics g)
    { paintOval(300, g); }
    /**
     * @param ovalWidth - gjerësia e elipsës
     * @param g - penda grafike */
    private void paintOval(int ovalWidth, Graphics g)
    { int backgroundSize = (int)(ovalWidth * scale);
      int backgroundHeight = (int)(2 * backgroundSize / 3);
      if ( backgroundHeight > 0 )
      { paintOval(backgroundSize, g);
      }
      int ovalHeight = (int)(2 * ovalWidth / 3);
      int top = (height - ovalHeight) / 2;
      g.setColor(Color.black);
      g.drawOval(0, top, ovalWidth, ovalHeight);
    }

    public static void main(String[] args)
    { new RecursiveOvalWriter(); }
  }
```

Për pyetjet 5–6 shqyrtoni metodën vijuese.

```
public int[] primes(int n)
{ int [] answer = new int[n];
  answer[0] = 2;
  answer[1] = 3;
  int count = 2;
  int next = 5;
  while ( count < n )
  { int limit = (int)(Math.sqrt(next));
    boolean found = false;
    int i = 1;
    while ( !found && answer[i] <= limit)
    { if ( next % answer[i] == 0 )
      { found = true; }
      else { i = i + 1; }
    }
    if ( !found )
    { answer[count] = next;
      count = count + 1;
    }
    next = next + 2;
  }
  return answer;
}
```

5. Çfarë kthen invokimi `primes(4)`?

   Përgjegjja: Vargun me termat `2,3,5,7`

6. Çfarë kthehet si rezultat i invokimit `primes(n)` për argumentin `n` numër të plotë më të madh se 1? Shkruani invariantat e të dy iterimeve në metodën. Shpjegoni algoritmin e metodës.

   Përgjegjja:

```
/** Kthen vargun e numrave të thjeshtë të gjatësisë sa argumenti i dhënë
 *  @param n - numri i dhënë natyror
 *  @return vargu i n numrave të parë të thjeshtë */
public int[] primes(int n)
{ int [] answer = new int[n];
  answer[0] = 2;
  answer[1] = 3;
  int count = 2;
  int next = 5;
  while ( count < n )
  // Vlerat e a[0],a[1],...,a[count-1] janë me rradhë count-1
  // numrat e parë të thjeshtë
  { int limit = (int)(Math.sqrt(next));
    boolean found = false;
    int i = 1;
    while ( !found && answer[i] <= limit)
    // Përderisa found == false,
    // asnjëri nga numrat e thjeshtë a[0],a[1],...,a[i-1]
    // nuk është pjesëtues i next
    { if ( next % answer[i] == 0 )
      { found = true; }
      else { i = i + 1; }
    }
    if ( !found )
    { answer[count] = next;
      count = count + 1;
    }
    next = next + 2;
  }
  return answer;
}
```

Algoritmi kërkon për vlerat teke duke filluar nga 5 të `next` se a kanë për pjesëtues ndonjërin nga numrat e thjeshtë (më të mëdhenjë se 2 e të cilët nuk e tejkalojnë $\sqrt{\texttt{next}}$) të gjetur më parë. Në qoftë se gjendet një pjesëtues, atëherë kalohet në vlerën vijuese të `next`; në të kundërtën vlera `next` ruhet si vlerë vijuese e thjeshtë në elementin vijues të vargut `answer`.

7. Çfarë do të afishojë urdhëri `while` vijues?

```
int number = 3000 ;
int i = 50 ;
while ( i <= 100  )
{
  if ( number % i == 0)
  { System.out.println(i); }
  i = i + 1;
}
```

Përgjegjja: Të gjithë pjesëtuesit e numrit 3000 të cilët ndodhen ndërmjet numrave 50 dhe 100 .