2|0|7|0|8|0|8|1|3

מספר התלמיד הנבחן
רשום את כל תשע הספרות

N101571590 ש

ת.ז: 207080813 מספר סידורי: 10

האוניברסיטה הפתוחה

י' בתמוז תש"ף

מס' שאלון - 478

2

ביולי 2020

סמסטר 2020ב

מס' מועד 81

20454 / 4

שאלון בחינת גמר

20454 - מבוא למדעי המחשב ושפת Java ב

משך בחינה: 3 שעות

בשאלון זה **12** עמודים

מבנה הבחינה:

קראו בעיון את ההנחיות שלהלן:
* בבחינה יש חמש שאלות.
* כל התכניות צריכות להיות מתועדות היטב.
יש לכתוב תחילה בקצרה את האלגוריתם וכל הסבר נוסף הדרוש להבנת התכנית.
יש לבחור בשמות משמעותיים למשתנים, לפונקציות ולקבועים שבתכנית.
תכנית שלא תתועד כנדרש לעיל תקבל לכל היותר % 85 מהניקוד.
* יש להקפיד לכתוב את התכניות בצורה מבנית ויעילה.
תכנית לא יעילה לא תקבל את מלוא הנקודות.
* אם ברצונכם להשתמש בתשובתכם בשיטה או במחלקה הכתובה בחוברת השקפים, אין צורך שתעתיקו את השיטה או את המחלקה למחברת הבחינה. מספיק להפנות למקום הנכון, ובלבד שההפניה תהיה מדויקת (פרמטרים, מיקום וכו').
* אין להשתמש במחלקות קיימות ב- Java , חוץ מאלו המפורטות בשאלות הבחינה.
* יש לשמור על סדר; תכנית הכתובה בצורה בלתי מסודרת עלולה לגרוע מהציון.
* בכתיבת התכניות יש להשתמש אך ורק במרכיבי השפה שנלמדו בקורס זה
אין להשתמש במשתנים גלובליים!
* את התשובות לשאלות 3 - 5 יש לכתוב על גבי השאלון. לא נבדוק תשובות שייכתבו במקום אחר!
* אפשר לתעד בעברית. אין צורך בתיעוד API .

חומר עזר:

ספר הלימוד : java software solutions מאת : lewis/loftus
חוברת השקפים של הקורס של ד"ר אמיר גורן ותמר וילנר.
יחידות 7-12. מותרות הערות בכתב יד, ע"ג הספרים.
אין להכניס חומר מודפס או כל חומר אחר מכל סוג שהוא.

בהצלחה !!!

החזירו
למשגיח את השאלון
וכל עזר אחר שקיבלתם בתוך מחברת התשובות

חלק א – עליכם לענות על כל השאלות בחלק זה במחברת הבחינה

שאלה 1 (25 נקודות)

צורף מכין שרשרת ארוכה ממקטעי שרשראות קצרות אותן הוא מחבר זו לזו. יש לו מקטעים באורכים שונים (סנטימטרים שלמים בלבד).

הוא רוצה ליצור שרשרת באורך k ס"מ. הוא יכול לקחת מכל אורך כמה מקטעים שירצה, אבל בסך הכל יכול לקחת num מקטעי שרשראות לכל היותר.

כתבו שיטה המקבלת כפרמטרים את אורך השרשרת הרצויה, את מספר המקטעים המקסימלי המותר, ואת האורכים השונים של המקטעים (מערך של מספרים שלמים שמהווים את האורכים). השיטה צריכה להחזיר מספר המציין בכמה אופנים הצורף יוכל ליצור את השרשרת באורך k.

לדוגמא, אם נתון ש:

- הסוגים השונים של מקטעי השרשראות הם באורכים (בסנטימטרים) 2 , 5, 10, 20, 50
- אורך השרשרת הרצויה הוא 40 ס"מ
- מותר לצורף לקחת עד 4 מקטעים בלבד

אזי האפשרויות להדבקת המקטעים הן:

- 20 + 20 (2 מקטעים)
- 20 + 10 + 10 (3 מקטעים)
- 20 + 10 + 5 + 5 (4 מקטעים)
- 10 + 10 + 10 + 10 (4 מקטעים)

ולכן התשובה שתוחזר מהשיטה תהיה 4.

חתימת השיטה היא:

```
public static int makeSum(int [] lengths, int k, int num)
```

המערך מכיל מספרים שלמים חיוביים (ממש) בלבד והוא אינו ממוין.

השיטה צריכה להיות רקורסיבית ללא שימוש בלולאות כלל. כך גם כל שיטות העזר שתכתבו (אם תכתבו) לא יכולות להכיל לולאות.

אפשר להשתמש בהעמסת-יתר (overloading).

אל תשכחו לתעד את מה שכתבתם!

```java
// 100    4 pieces only
public static int makeSum(int [] lengths,int k,int num) {
    return makeSum(lengths,k,num,0);
}

private static int makeSum(int [] lengths,int k,int num,int i) {

    if(num < 0 || k < 0 || i == lengths.length)
        return 0;

    if (k == 0 && num >= 0)
        return 1;

    return makeSum(lengths,k-lengths[i],num-1,i) + makeSum(lengths,k,num,i+1);
}
```

שאלה 2 (25 נקודות)

נתון מערך חד-ממדי a המלא במספרים שלמים.

כתבו שיטה סטטית, המקבלת כפרמטר מערך כזה, ומספר שלם k. השיטה צריכה למצוא את התת-מערך בגודל k שסכומו הוא מינימלי, ולהדפיס את האינדקסים של תחילת התת-מערך ושל סופו. ניתן להניח כי k אינו גדול מגודל המערך.

לדוגמא,

אם המערך a הוא זה:

0	1	2	3	4	5	6	7	8	9
10	4	2	5	6	3	8	1	5	9

- אם המספר k = 3

אז התת-מערך בגודל 3 שסכומו הוא הקטן ביותר הוא זה שמתחיל באינדקס 1 ומסתיים באינדקס 3. הסכום 5 + 2 + 4 הוא הקטן ביותר.

השיטה תדפיס Minimum sum sub-array is (1,3)

- אם המספר k = 2

אז יש שני תת-מערכים בגודל 2 שסכומם הוא הקטן ביותר (6). גם זה המתחיל באינדקס 1 ומסתיים באינדקס 2, וגם זה המתחיל באינדקס 7 ומסתיים באינדקס 8. לכן השיטה תדפיס **אחד** מהם באופן שרירותי.

השיטה תדפיס Minimum sum sub-array is (1,2) **או** Minimum sum sub-array is (7,8)

חתימת השיטה היא:

```
public static void minimumSubK (int[] arr, int k)
```

שימו לב:

השיטה שתכתבו צריכה להיות יעילה ככל הניתן, גם מבחינת סיבוכיות הזמן וגם מבחינת סיבוכיות המקום. תשובה שאינה יעילה מספיק כלומר, שתהיה בסיבוכיות גדולה יותר מזו הנדרשת לפתרון הבעיה תקבל מעט נקודות בלבד.

מה סיבוכיות זמן הריצה וסיבוכיות המקום של השיטה שכתבתם? הסבירו תשובתכם.

אל תשכחו לתעד את מה שכתבתם!

```java
public static void minimumSubK(int [] arr,int k) {
    int startIndex = 0;
    int stopIndex = k-1;
    int sum = 0;
    int lastSum = 0;
    int i;
    for(i=0; i<k;i++) {
        sum+=arr[i];
    }
    lastSum=sum;
    for(i=k; i<arr.length;i++) {
        sum+=arr[i]-arr[i-k];
        if(sum<lastSum) {
            lastSum=sum;
            stopIndex = i;
            startIndex = i-k+1;
        }
    }
    System.out.println("Minimum sum sub-array is" + "(" +startIndex + "," + stopIndex + ")" );
}
```

חלק ב - את התשובות לשאלות 3- 5 יש לכתוב על גבי השאלון.
לא נבדוק תשובות שייכתבו במקום אחר!

שאלה 3 (17 נקודות)

נניח שהמחלקה Node שלהלן מממשת עץ בינרי.

```
public class Node
{
     private int _number;
     private Node _leftSon, _rightSon;

     public Node (int number)
     {
        _number = number;
        _leftSon = null;
        _rightSon = null;
     }

     public int  getNumber()       {return _number; }
     public Node getLeftSon()      {return _leftSon; }
     public Node getRightSon()     {return _rightSon; }
}
```

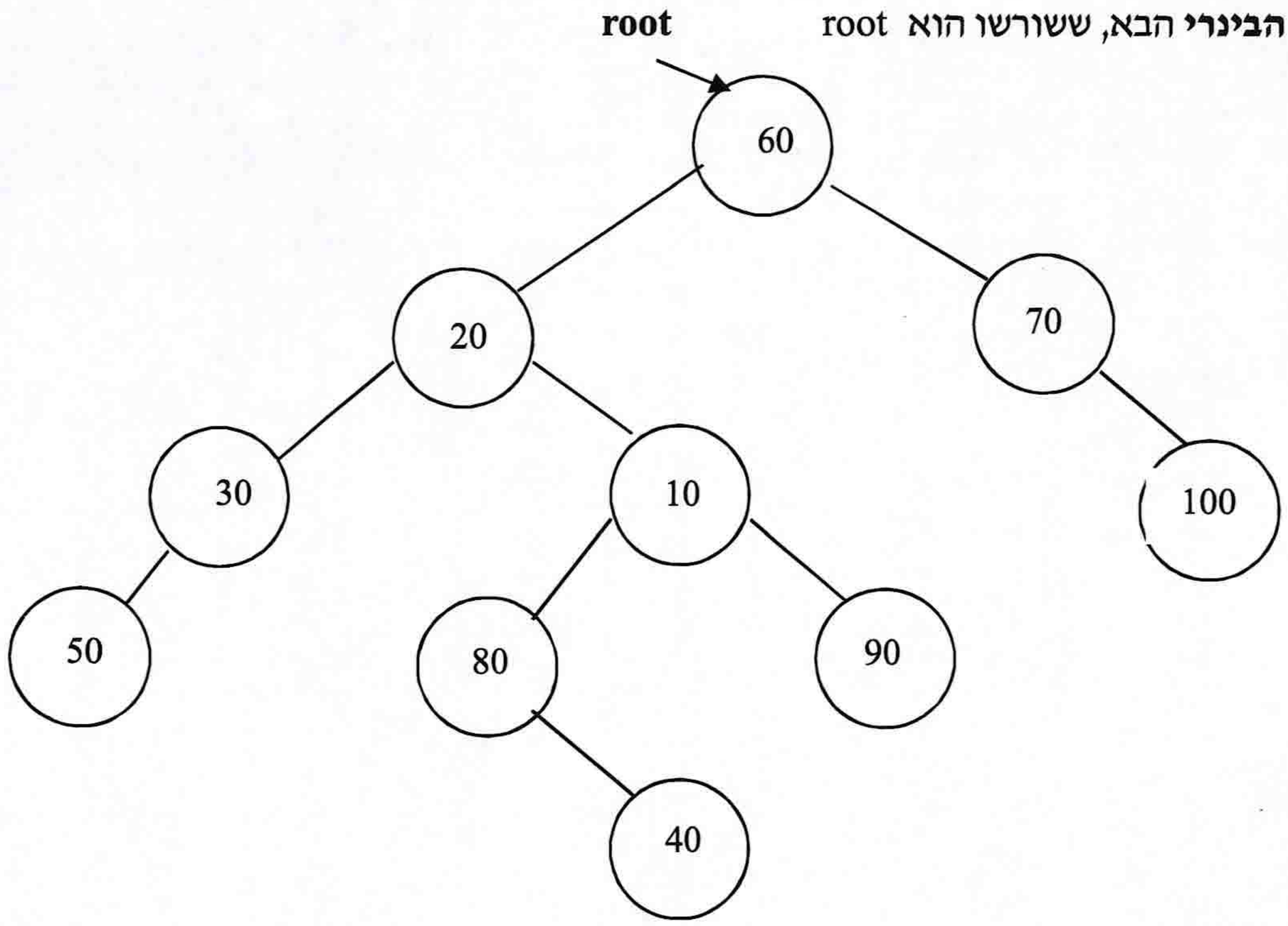
המחלקה BinaryTree מאגדת בתוכה שיטות סטטיות לטיפול **בעץ בינרי**.

בין השיטות נתונות השיטות what ו- whatBool הבאות:

```
     private static int what(Node root)
     {
         if (root == null )
             return 0;
         int num1 = what(root.getLeftSon());
         int num2 = what(root.getRightSon());
         if (Math.abs(num1 - num2) > 1)
             return 0;
         return Math.max(num1, num2) + 1;
     }

     public static boolean whatBool(Node root)
     {
         return (what(root) >0);
     }
```

נתון **העץ הבינרי** הבא, ששורשו הוא root



ענו על הסעיפים הבאים:

(4 נק') (א) איזה ערך תחזיר השיטה **whatBool** בעקבות הקריאה BinaryTree.whatBool(root)?

התשובה היא:

false

(5 נק') (ב) איזה שינוי **מינימלי** אפשר לעשות **על העץ** לעיל, כדי שהקריאה לשיטה whatBool(root) תחזיר ערך אחר מזה שהוחזר בסעיף א? **שימו לב, השינוי צריך להיות בעץ ולא בשיטה (כגון, הוספת צומת, הורדת צומת, שינוי ערך של צומת וכד'). מינימלי במובן של מינימום פעולות על העץ.** כך, אם מורידים צומת, ויש לצומת הזה בנים, במספר הפעולות נספרים גם הבנים של הצומת שהורדו.

התשובה היא:

ניתן להוסיף צומת נוספת מצד ימין של הערך 100 ,
או לחליפין להסיר צומת נניח את 40.

(8 נק') (ג) מה מבצעת השיטה **whatBool** באופן כללי כשהיא מקבלת כפרמטר שורש של **עץ בינרי** root? שימו לב, עליכם לתת תיאור ממצה של **מה עושה השיטה באופן כללי**, ולא תיאור של מה עושה כל שורה בשיטה, או איך היא מבצעת זאת. **כלומר, מה המשמעות של הערך שהשיטה מחזירה?**

התשובה היא:

השיטה בודקת את את עומק תתי העצים של צד ימין וצד שמאל.
אם ההפרש ביניהם גדול מ-1 יחזיר False אחרת יחזיר True.

שאלה 4 (15 נקודות)

נתון פרויקט שהוגדרו בו המחלקות First, Second, Third שלהלן. כל אחת בקובץ נפרד, כמובן.

```
public class First
{
    public static int count = 0;
    private String _str;

    public First() {
        count++;
        _str = "Empty " + count;
    }

    public First(String str) {
        count++;
        _str = str;
    }

    public First(First g) {
        count++;
        _str = "Copy" + g._str;
    }

    public void setStr(String str) {
        _str = str;
    }

    public void print() {
        System.out.println("First " + _str);
    }
}

//-----------------------------
public class Second extends First
{
    private First _f;

    public Second(First f1, First f2) {
        super(f1);
        _f = f2;
    }

    public void setStr(String str) {
        super.setStr(str);
        _f = new First(str);
    }

    public void print() {
        System.out.println("Second");
        super.print();
        _f.print();
    }
}
```

```
public class Third
{
    private int _curr;
    private First[] _arr;

    public Third(int count)
    {
        _curr = 0;
        _arr = new First[count];
    }

    public void print()
    {
        for (int i = 0; i < _curr; i++)
            _arr[i].print();
    }

    public void add(First s)
    {
        if (_curr < _arr.length) {
            _arr[_curr] = s;
            _curr++;
        }
    }
}
```

בפרויקט נמצאת גם המחלקה Driver ובה השיטה main.

בסעיפים הבאים כתבו מה יודפס על הפלט לאחר ביצוע הקוד שבסעיף. הניחו שקטעי הקוד נכתבים אחד אחרי השני בשיטה main. כלומר הקטעים מצטברים זה אחר זה, ולא מופיעים כל אחד בנפרד. **כתבו את הפלט של השורות שנוספו בלבד, אין לכתוב את הפלט של השורות הקודמות.**

לא בהכרח כל השורות יתמלאו.

1.
```
First f1 = new First("One");
f1.print();
```

First One

2.
```
First f2 = new First("Two");
f2.print();
```

First Two

3. `First f3 = new First(f2);`
`f3.print();`

First Copy Two

4. `First f4 = new First();`
`f4.print();`

First Empty 4

5. `Second s1 = new Second(f1, f3);`
`s1.print();`

Second
First Copy One
First copy Two

6. `Second s2 = new Second(f4, f2);`
`s2.print();`

Second
First copy Empty 4
First Two

7.
```
Third t = new Third(First.count);
t.add(f1);
t.add(f2);
t.add(s1);
t.add(s2);
System.out.println(First.count);
t.print();
```

```
6
First One
First Two
Second
First copy one
First copy two
second
first copy Empty 4
First Two
```

8.
```
f1.setStr("Five");
s2.setStr("Six");
System.out.println(First.count);
t.print();
```

```
7
First five
first two
second
first copy one
first copy two
second
first six
first six
```

שאלה 5 (18 נקודות)

נתונה המחלקה IntNode הבאה, המייצגת איבר ברשימה מקושרת חד-סטרית בה יש מצביע לאיבר הבא, המכילה מספרים שלמים:

```
public class IntNode
{
    private int _value;
    private IntNode _next;

    public IntNode(int val, IntNode n) {
        _value = val;
        _next = n;
    }

    public IntNode getNext( )  {
          return _next;
    }

    public int getValue()  {
          return _value;
    }
}
```

נתונה רשימה מקושרת **חד-סטרית**, הממומשת בעזרת המחלקה IntList שלהלן:

```
public class IntList
{
    private IntNode _head;
    public IntList()
    {
        _head = null;
    }

                // כאן יש עוד בנאים ושיטות...  //
                //    המשך המחלקה בעמוד הבא    //
```

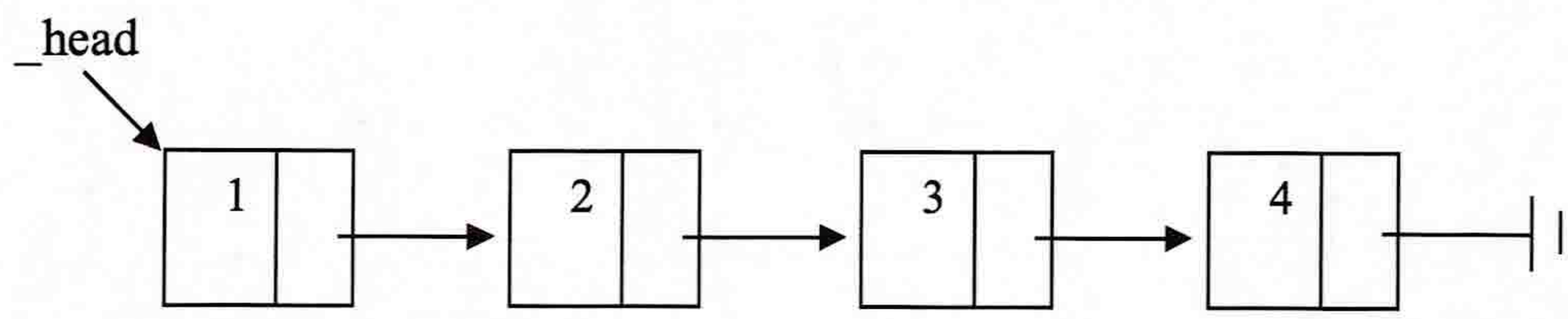
```
public int what(int x, int y)
    {
        int p1 = -1, p2 = -1, d = Integer.MAX_VALUE;
        int i = 1;
        IntNode ptr = _head;
        while (ptr!=null)
        {
(1)         if (ptr.getValue() == x)
            {
                p1 = i;
                if (p2 != -1)
                    d = Math.min(d, Math.abs(p1 - p2));
            }
(2)         else if (ptr.getValue() == y)
            {
                p2 = i;
                if (p1 != -1)
                    d = Math.min(d, Math.abs(p1 - p2));
            }
            ptr = ptr.getNext();
            i++;
        }
        return d;
    }

    . . .                // other methods

}    // end of class IntList
```

הניחו שיש במחלקה גם שיטה שמכניסה ערכים לרשימה. אינכם צריכים לדאוג לכך.

בשאלות להלן, נסמן את איברי הרשימה כמספרים מופרדים בפסיקים, בתוך סוגריים מסולסלים. כך לדוגמא, נסמן { 1 , 2 , 3 , 4 } את הרשימה שלהלן (המספר הראשון משמאל הוא המספר שבראש הרשימה):



סעיף א (4 נקודות)

בהנחה שהרשימה list עליה נפעיל את השיטה what היא זו:

list = { 20, 50, 80, 40, 50, 30, 50, 20, 40, 30 }

איזה ערך תחזיר הקריאה לשיטה list.what(20, 30) ?

התשובה היא

2

סעיף ב (4 נקודות)

נניח שנשנה את השורות הממוספרות ב- (1) ו- (2) בשיטה לעיל כך שיהיו:

```
(1)          if (ptr.getValue() <= x)

(2)          else if (ptr.getValue() >= y)
```

בהנחה שהרשימה list עליה נפעיל את השיטה what היא זו:

list = { 20, 50, 80, 40, 50, 30, 50, 20, 40, 30 }

איזה ערך תחזיר הקריאה לשיטה list.what(20, 30) ?

התשובה היא

1

סעיף ג (10 נקודות)

מה מבצעת השיטה **what** (המקורית, לפני השינוי) באופן כללי? הסבירו בקצרה **מה** השיטה עושה ולא **כיצד** היא מבצעת זאת.

שימו לב, עליכם לתת תיאור ממצה של **מה עושה השיטה באופן כללי**, ולא תיאור של מה עושה כל שורה בשיטה, או איך היא מבצעת זאת. כלומר, עליכם לכתוב מה המשמעות של הערך שמוחזר מהשיטה what, כשהיא מופעלת על רשימה כלשהי עם פרמטרים x ו-y כלשהם? **התייחסו למקרי קצה.**

התשובה היא:

- השיטה מחזירה את המרחק הקטן ביותר בין 2 מופעים של x ו y ברשימה.
- אם אחד משני המספרים לא קיימים ברשימה , יוחזר הערך המקסימלי ביותר שקיים בJava .

בהצלחה!