```
using ListProj;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ListProj
{
    /*~~~~~~~~~~~~~~~~~~~~~  פעולות עזר לרשימה ~~~~~~~~~~~~~~~~~~~
     *                פעולות חישוב ברשימה
     *             הוספה ומחיקה של איברים
     *                      חיפוש ברשימה
     *        החזרת הפניות לאיברים ברשימה
     *                  בנייה והצגת רשימה
     *
     *
     *         הערה חשובה: כל הפעולות עוסקות ברשימה של מספרים שלמים
     *  יש להצע התאמות לטיפוס הנתונים במקרה של רשימה של
     *                            מספרים שלמים או תווים
     *                      במקרה של מחרוזות או עצמים,
     *                    Equals - יש לזכור שמשווים ב
     *~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
     */
    internal class List
    {
        public static void Main(string[] args)
        {
            int[] arr = { 4, 7, 2, 9, 1, -5, 12 };
            Node<int> lst1 = BuildList(arr);
            Show(lst1);

            Node<int> lst2 = GetList(arr);
            Show(lst2);
        }

        //================================
        //===== פעולות חישוב ברשימה  =====
        //================================

        //--- פעולה המחזירה את מספר האיברים ברשימה ---
        public static int Size(Node<int> lst)
        {
            int count = 0;
            while (lst != null)
            {
                count++;
                lst = lst.GetNext();
            }
            return count;
        }


```

```
54
55          //--- פעולה המחזירה את סכום איברי הרשימה ---
56          public static int SumLst(Node<int> lst)
57          {
58              int sum = 0;
59              while (lst != null)
60              {
61                  sum += lst.GetValue();
62                  lst = lst.GetNext();
63              }
64              return sum;
65          }
66
67          //--- פעולה המחזירה את הערך הגדול ביותר ברשימה ---
68          //---                    הנחה: הרשימה לא ריקה ---
69          public static int MaxValue(Node<int> lst)
70          {
71              int max = int.MinValue;
72              int x;
73              while (lst != null)
74              {
75                  x = lst.GetValue();
76                  if (x > max)
77                      max = x;
78                  lst = lst.GetNext();
79              }
80              return max;
81          }
82
83          //---  פעולה המחזירה את הערך הקטן ביותר ברשימה ---
84          //---                    הנחה: הרשימה לא ריקה ---
85          public static int MinValue(Node<int> lst)
86          {
87              int min = int.MaxValue;
88              int x;
89              while (lst != null)
90              {
91                  x = lst.GetValue();
92                  if (x < min)
93                      min = x;
94                  lst = lst.GetNext();
95              }
96              return min;
97          }
98
99          //---             פעולה המקבלת רשימה ומספר שלם ---
100         //--- ומחזירה רת מספר מופעיו של מספר זה ברשימה ---
101         public static int CountX(Node<int> lst, int x)
102         {
103             int count = 0;
104             while (lst != null)
105             {
106                 if (lst.GetValue() == x)
```

```
107                     count++;
108                 lst = lst.GetNext();
109             }
110             return count;
111         }
112 
113         //================================
114         //==== הוספה ומחיקה של איברים ====
115         //================================
116 
117         //--- הוספת חוליה בצורה ממויינת לרשימה ממויינת ---
118         public static Node<int> InserSorted(Node<int> lst, int x)
119         {
120             Node<int> pos = lst, prev = null;
121             while (pos != null && pos.GetValue() < x)
122             {
123                 prev = pos;
124                 pos = pos.GetNext();
125             }
126             if (prev == null)
127                 lst = new Node<int>(x, lst);
128             else
129                 prev.SetNext(new Node<int>(x, pos));
130             return lst;
131         }
132 
133         //---              x פעולה המוסיפה חוליה חדשה בערך ---
134         ///---           null שאינו  lst ברשימה pos במקום ---
135         //--- אם הערך הוא null יתווסף המספר בתחילת הרשימה ---
136         public static Node<int> Add(Node<int> lst, Node<int> pos, int x)
137         {
138             if (pos == null)        //  הוספה בתחילת הרשימה
139                 lst = new Node<int>(x, lst);
140             else
141                 pos.SetNext(new Node<int>(x, pos.GetNext()));
142 
143             return lst;
144         }
145 
146         //--- פעולה המקבלת מספר ומוחקת אותו מהרשימה ---
147         //--- אם x לא קיים ברשימה - לא עושים דבר ---
148         public static Node<int> Remove(Node<int> lst, int x)
149         {
150             if (lst.GetValue() == x)        //  מחיקת הראשון
151                 return lst.GetNext();
152 
153             Node<int> prev = lst;
154             Node<int> pos = prev.GetNext();  // pos = lst.GetNext(); או
155 
156             while (pos != null && pos.GetValue() != x)
157             {
158                 prev = pos;
159                 pos = pos.GetNext();
```

```
160             }
161
162             if (pos != null)        // המשמעות היא  x לא קיים ברשימה
163                 prev.SetNext(pos.GetNext());
164
165             return lst;
166         }
167
168         //--- פעולה המקבלת מקום ברשימה ומוחקת אותו ---
169         //--- null הנחה: pos הוא מקום ברשימה שאינו ---
170         public static Node<int> Remove(Node<int> lst, Node<int> pos)
171         {
172             Node<int> prev = GetPrev(lst, pos);
173             prev.SetNext(pos.GetNext());
174             return lst;
175         }
176
177
178
179
180
181         //=========================================
182         //=====           חיפוש ברשימה        =====
183         //=========================================
184         //--- פעולה המקבלת רשימה ומספר ומחזירה  ---
185         //--- אמת אם המספר קיים ברשימה ושקר אחרת ---
186         public static bool Exist(Node<int> lst, int x)
187         {
188             while (lst != null)
189             {
190                 if (lst.GetValue() == x)
191                     return true;
192                 lst = lst.GetNext();
193             }
194             return false;
195         }
196
197         //---            פעולה המקבלת רשימה ומספר שלם ---
198         //--- אם המספר קיים ברשימה, תוחזר הפנייה אליו ---
199         //---                     null אחרת יוחזר ערך ---
200         public static Node<int> FindX(Node<int> lst, int x)
201         {
202             while (lst != null && lst.GetValue() != x)
203                 lst = lst.GetNext();
204             return lst;
205         }
206
207
208
209
210
211
212
```

```
213         //=========================================
214         //====   פעולות המחזירות הפניות ברשימה ====
215         //=========================================
216
217         //--- lst ברשימה pos -פעולה המחזירה הפנייה לחולייה הקודמת ל ---
218         //--- null מצביע על החוליה הראשונה ברשימה, יוחזר ערך pos אם ---
219         //---                  null הוא מקום ברשימה שאינו pos :הנחה ---
220         public static Node<int> GetPrev(Node<int> lst, Node<int> pos)
221         {
222             Node<int> prev = null;
223             while (lst != pos)
224             {
225                 prev = lst;
226                 lst = lst.GetNext();
227             }
228             return prev;
229         }
230
231         //--- פעולה המקבלת רשימה ומחזירה הפניה לחוליה האחרונה בה ---
232         public static Node<int> GetLast(Node<int> lst)
233         {
234             while (lst.HasNext())
235                 lst = lst.GetNext();
236             return lst;
237         }
238
239
240
241
242
243
244         //=================================
245         //=====  בנייה והצגת רשימה   =====
246         //=================================
247
248         public static Random rnd = new Random();
249
250         //--- בניית רשימה ראנדומלית של מספרים מהסוף להתחלה
251         public static Node<int> GetRandomList(int size, int from, int to)
252         {
253             Node<int> lst = null;
254             int x;
255             for (int i = 0; i < size; i++)
256             {
257                 x = rnd.Next(from, to + 1);
258                 lst = new Node<int>(x, lst);
259             }
260
261             return lst;
262         }
263
264
265
```

```
266         //--- פעולה היוצרת רשימה של מספרים מהקלט ---
267         //---    סדר האיברים ברשימה תואם את הקלט ---
268         //---                  שימוש בחוליית דמה ---
269         public static Node<int> BuildList()
270         {
271             Node<int> lst = new Node<int>(-999);
272             Node<int> pos = lst;
273 
274             Console.Write("type 1st number (-1 to finish) --> ");
275             int num = int.Parse(Console.ReadLine());
276             while (num != -1)
277             {
278                 pos.SetNext(new Node<int>(num));
279                 pos = pos.GetNext();
280 
281                 Console.Write("type next number --> ");
282                 num = int.Parse(Console.ReadLine());
283             }
284             return lst.GetNext();
285         }
286 
287         //--- בניית רשימה מהסוף להתחלה ---
288         public static Node<int> BuildList(int[] arr)
289         {
290             Node<int> lst = null;
291             for (int i = 0; i < arr.Length; i++)
292                 lst = new Node<int>(arr[i], lst);
293             return lst;
294         }
295 
296         //--- בניית רשימה מההתחלה לסוף ---
297         public static Node<int> GetList(int[] arr)
298         {
299             Node<int> lst = new Node<int>(arr[0]);
300             Node<int> pos = lst;
301             for (int i = 1; i < arr.Length; i++)
302             {
303                 pos.SetNext(new Node<int>(arr[i]));
304                 pos = pos.GetNext();
305             }
306             return lst;
307         }
308 
309 
310 
311 
312 
313 
314 
315 
316 
317 
318 
```

```
319         //---   הדפסת איברי הרשימה בשורה ---
320         public static void Show(Node<int> lst)
321         {
322             Console.Write("[");
323             Node<int> pos = lst;
324             while (pos != null)
325             {
326                 Console.Write(pos.GetValue());
327                 if (pos.HasNext())
328                     Console.Write(", ");
329                 pos = pos.GetNext();
330             }
331             Console.WriteLine("]");
332         }
333 
334         //--- הדפסת איברי הרשימה כולל שם הרשימה ---
335         public static void Show(string str, Node<int> lst)
336         {
337             Console.Write(str + ": [");
338             Node<int> pos = lst;
339             while (pos != null)
340             {
341                 Console.Write(pos.GetValue());
342                 if (pos.HasNext())
343                     Console.Write(", ");
344                 pos = pos.GetNext();
345             }
346             Console.WriteLine("]");
347         }
348     }
349 }
350 
351 
```