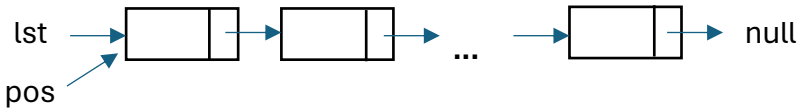
אילת וייץ

בס"ד

רשימות מקושרות

בהסבר הבא נתבסס על קטע הקוד הבא:

```
Node<int> pos=lst;
```



(pos הוא מצביע עזר לרשימה קיימת בלשהיא lst)

<u>מעבר על לולאה:</u>

<u>סריקה:</u>

```
while(pos!=null)
```

<u>הוספה:</u>

```
Node<int> addPos=new Node<int>(_);
```

הוספה מתחלקת להוספה לרשימה/אמצע או סוף רשימה.

הוספה לראש רשימה:

```
addPos.SetNext(lst);

lst=addPos;
```

הוספה לאמצע/סוף רשימה

כשנרצה להוסיף חוליה לאמצע רשימה, נצטרך להצביע על החולייה שנרצה להוסיף את החולייה החדשה אחריה, לשנות את התכונה next שלה להצביע על החולייה החדשה, ואת החולייה החדשה לחבר לתכונה next של החולייה עליה עמדנו.

נדגים:

```
addPos.SetNext(pos.Getnext());

pos.setNext(addPos);
```

יש חשיבות לסדר!

למה?

מכיוון שאם קודם נקשר את החולייה עליה נעמוד לחוליה החדשה, אח"כ נרצה לקשר את החולייה החדשה ל pos.GetNext() שהיא החולייה אליה הצביעה עד עתה pos, כבר לא נוכל-כי pos כבר יצביע לחוליה החדשה...

אם נצטרך לבצע הוספה <u>לפני</u> חוליה מסוימת, נצטרך לסרוק באמצעות

```
while(pos.GetNext()!=null)
```

כשכל פעם הבדיקה היא על האיבר הבא.

לעומת זאת אם ההוספה היא לאחר חוליה מסוימת, נצטרך לסרוק באמצעות

while(pos!=null)

הוספה לסוף רשימה

נעבור בלולאה עד לסוף הרשימה, אך נרצה לעצור כאשר ה pos יצביע על החולייה האחרונה, טרם יגיע ל null. כי לא ניתן להוסיף חוליה אשר null יצביע עליה...

ולכן:

```
while(pos.GetNext()!=null)
{
    pos=pos.GetNext();
}
pos.SetNext(addPos);
```

(כאן אין צורך לשנות את התכונה next של החולייה החדשה, כי היא האחרונה ותמשיך להצביע ל null)

כשלא ידוע מראש המצביע/מיקום (התחלה/אמצע/סוף) אליו נצטרך להוסיף חוליה חדשה, נצטרך לבדוק את כל האופציות.

אם מדובר בהוספה לפני חוליה מסוימת (לדו' נתונה רשימה, לפני כל מספר זוגי יש להוסיף מספר אי זוגי הקטן ממנו ב 1)

נוכל לדעת שבוודאות אין כאן הוספה לסוף הרשימה, כך שנצטרך לבדוק ראש/אמצע רשימה

נבדוק מחוץ ללולאה אם מדובר בהוספה לאיבר הראשון, ונבצע הוספה במקרה הצורך.

לדו':

```
If(lst.GetValue()%2==0)
{
Node<int> addPos=new Node<int>(lst.GetValue()-1);
addPos.SetNext(lst);
lst=addPos;
}
```

ועל שאר הרשימה נעבור באמצעות:

while(pos.GetNext()!=null)

כאשר בכל פעם הבדיקה תהיה על האיבר הבא, על מנת שבמקרה שנצטרך להוסיף חוליה לפניה נוכל לחברה ל pos.

אם מדובר בהוספה לאחר חוליה מסוימת (לדו' נתונה רשימה, אחרי כל מספר זוגי יש להוסיף מספר אי זוגי הקטן ממנו ב 1)

נוכל לדעת בוודאות שאין כאן הוספה לראש הרשימה, ולכן נעבור בלולאה עבור הוספה לאמצע/סוף

נעבור על הרשימה באמצעות:

while(pos!=null)

מכיוון שכל הוספה תהיה חיבור חוליה חדשה ל pos. את החוליה החדשה נחבר ל pos.GetNext() ולאחר מכן נשנה את התכונה next של החוליה החדשה ל pos.

<u>מחיקה מרשימה</u>

הסרה מראש רשימה

נצטרך לשנות את המצביע לראש הרשימה למצביע הבא. במקרה שלא ידוע כמה איברים יש למחוק מראש הרשימה, נעבור בלולאה עד שנגיע לאיבר שלא צריך להסיר מהרשימה, והמצביע לראש הרשימה יצביע אליו.

לדו':

while(lst!=null&& _____*)

{

lst=lst.GetNext();

}

*התנאי למחיקה

כך איבדנו את הגישה לאיברים ש"נמחקו" (זו מחיקה רדודה, במחיקה עמוקה נשמור בכל פעם את המצביע לחוליה למחיקה במשתנה עזר, ולאחר קידום המצביע נשנה את ערך התכונה next של מצביע העזר ל null)

הסרה מאמצע/סוף רשימה

כשמסירים איבר מרשימה, טרם ההסרה נצטרך לחבר את החולייה לפני החולייה להסרה אם זו שלאחריה. ולכן הבדיקה האם יש להסיר חוליה תתבצע כך שבכל מעבר על חוליה נבדוק את זו שאחריה. כך: (במקרה שלפני כן בוצעה הסרה מראש הרשימה, יש לעדכן pos=lst;)

while(pos.GetNext()!=null)

<u>בד"כ בפעולה המשנה את הרכב הרשימה (מסירה/מוסיפה) נחזיר את ראש הרשימה החדש!</u>