1.

Cevap ArrayListtir. Arraylist veri yapısında sona ekleme işlemi sabit zamandadır yani maliyetsizdir. Arraylist sınıfında bir data dizisi tuttuğumuz için diziyi komple silmek istediğimizde tek tek elemanları dolaşmak yerine delete[] data dememiz yeterli olur. Ayrıca arraylistte eleman eklenecek indexi sürekli güncel tuttuğumuz için indexi döndürmek her zaman bize dizinin boyutunu verir bu da kolay bir işlemdir.

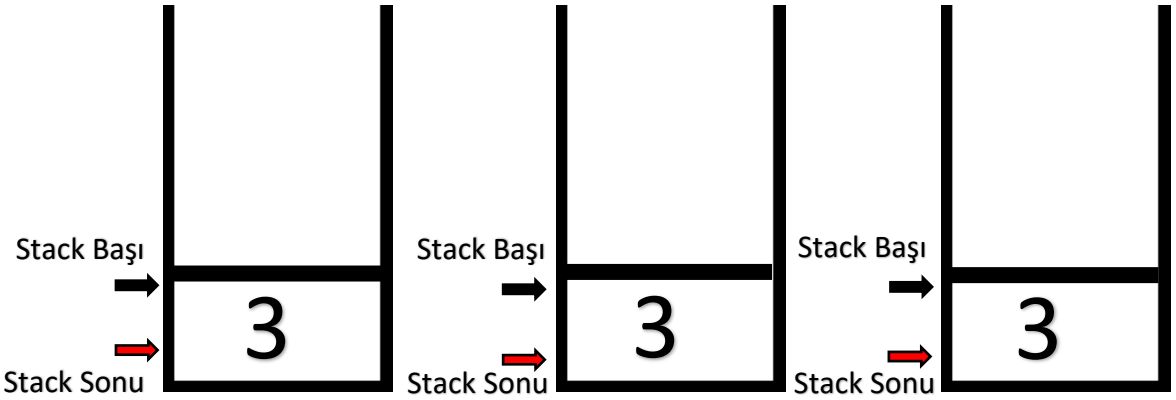
2.

```
void soldanEkle(const int& deger)
{
    LB = new Dugum(deger, LB, LB->geri);
    LB->ileri->geri = LB;
    LB->geri->ileri = LB;
}
```

3.

Numara 33 olduğu varsayılırsa;

x = 3 ve y = 3 → y-- → x = 3 ve y = 2 → y-- → x = 3 ve = y =1 → y-- ve y=0



4.

Okul no sonu 3 olduğu düşünülürse infix ifade **(5 – 4 + 6) * (8 / (10 + 2)) / 5** 'tir.

Postfix dönüşümü ise aşağıdadır;

Infix: (5 – 4 + 6) * (8 / (10 + 2)) / 5

Infix to Postfix

Postfix: 5 – 4 6+ 8 10 2+/ * 5/

Step by step output for '' expression

Input String	Output Stack	Operator Stack
(5 – 4 + 6) * (8 / (10 + 2)) / 5		(
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5	(
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5	(
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 –	(
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 –	(
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4	(
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4	(
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4	(+
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4	(+
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6	(+
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+	
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+	
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+	*
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+	*
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+	*(
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8	*(
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8	*(
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8	*(/
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8	*(/
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8	*(/(
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8 1	*(/(
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8 10	*(/(
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8 10	*(/(
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8 10	*(/(+
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8 10	*(/(+
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8 10 2	*(/(+
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8 10 2+	*(/
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8 10 2+/	*
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8 10 2+/	*
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8 10 2+/ *	/
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8 10 2+/ *	/
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8 10 2+/ * 5	/
(5 – 4 + 6) * (8 / (10 + 2)) / 5	5 – 4 6+ 8 10 2+/ * 5/	

Learn how to convert infix to postfix.

5.

```
void insertlist(int konum, ArrayList<Object> &list)
{
   for(int i=0; i<list.length(); i++)
   {
      insert(konum++, list.Elemanlar[i]);
   }
}
```