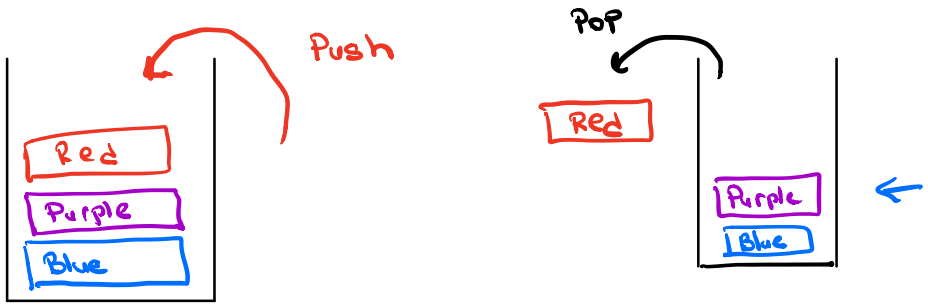
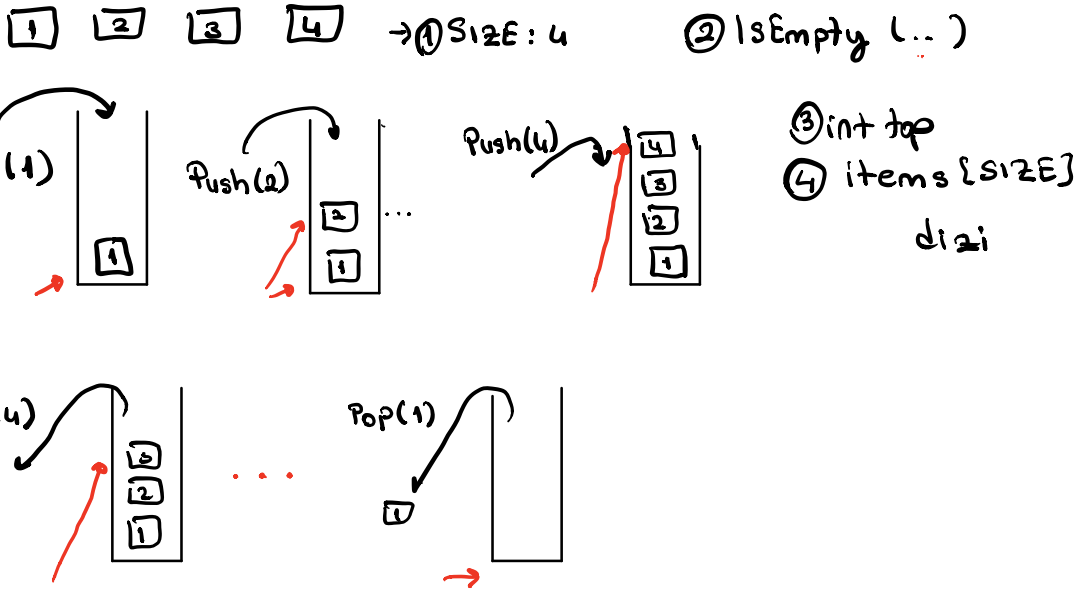
(Stack) Yığıt Veri Yapısı : Sadece tek yönden hareketi olan dizeye (stack) yığıt denir.





Örnek Program: (Struct) Veri yapısını belirleyelim:

```
#define SIZE 4

Yığıt Veri Yapısı
struct stack {
    float items[SIZE];
    int top; };
```

Syntax:
Push : ++ s->top
x: s->items[s->top]
POP:
s->items[-- s->top]

Test Fonksiyonu:

```
bool IsEmpty (struct stack *s)
{   if (s->top==-1)
        return true;
```

→ pointer'ın gösterdiği sayıcı, -1 ise true değilse false gönderir.

```
    else
        return false;
}
```

Yığıt içinden Eleman çıkaran Fonksiyon: (POP)

```
float Pop(struct stack *s)
{   if( !IsEmpty(s))
    {   double x = s->items[s->top];
        (s->top)--;
        return x;
    // return s->items[s->top]  <- deneme!
    }
    else
    {  printf("yigit bos, islem yapilamadi\n");
       return 0;
    }
}
```

Yığıt içine Eleman ekleyen Fonksiyon;

```
void Push( struct stack *s, double x)
{
    if (s->top < SIZE-1)
    {   (s->top)++;
        s->item[s->top] = x;
    }
    else
    } printf("yigit dolu.\n");
}
```

Ekrana yazdıralım (Pek akıllıca olmayan yöntem)

```
void PrintStack ( struct stack *s)
{   while ( ! IsEmpty (s))
      printf ("%.2.1 f", pop(s) )
}
```

→ noktadan sonra tek basamak yazdır.

ANA PROGRAM

```
int main ( )
{
   double d;
   struct Stack y;
   y. top = -1;
   push ( &y , 1);
   push (&y , 2.779);
   d = pop( &y);
   printf ( "yigitten cekilen ekman
                          = %.2.1f \n ", d );
   d = pop(&y);
   return 0;
}
```

→ silinen elemanı atayacağım değişken

→ yığıt veri yapısı

→ 1 ekle

→ 2.779 ekle

→ LIFO 'ya göre sil.