Özyineleme (Recursion)

Bir fonksiyon veya metodun kendi kendini çağırma işlemine özyineleme denir. Özyinelemenin 2 temel kuralı vardır.
- Sonlanacağı durum (Temel adım)
- Her çağrıda sonlanacağı duruma yaklaşması (Özyineleme)

Eğer bir sonlanma durumu olmaz ise sonsuz çağrım olacak ve program hata verecektir. Sonlanma durumu var ama her çağrıda o duruma yaklaşılmaz ise yine sonsuz çağrım olur. Özyineleme birçok problemde kod satırlarının, bir iki satıra inmesini sağlar.

Toplam

```
int Toplam(unsigned int sayi){
	if(sayi == 0) return 0;
	return sayi + Toplam(sayi-1);
}
```

Faktöriyel

```
int Fakt(unsigned int sayi){
	if(sayi == 0) return 1;
	return sayi * Fakt(sayi-1);
}
```

Fibonacci

```
int Fib(unsigned int sayi){
	if(sayi == 1 || sayi == 2) return 1;
	return Fib(sayi-1) + Fib(sayi-2);
}
```

Asal Kontrolü

```
bool Asal(int sayi, int bolen=2){
	if(sayi == bolen || sayi == 1) return true;
	if(sayi % bolen == 0) return false;
	return Asal(sayi,bolen+1);
}
```

1'den girilen sayıya kadar ekrana yazılması

```
void Yaz(int sayi){
	if(sayi==0)return;
	Yaz(sayi-1);
	cout<<sayi<<" ";
}
```

Us

```
int Us(int sayi,int us)
{
  if(us==0)return 1;
  return sayi*Us(sayi,us-1);
}
```

Şablonlar

Şablonlar türden bağımsız fonksiyonlar tanımlamayı sağlarlar.

```
#include <iostream>
using namespace std;

template <typename Nesne>
void Karsilastir(Nesne x,Nesne y) {
	if(x>y)cout<<"Buyuk";
	else if(x<y)cout<<"Kucuk";
	else cout<<"Esit";
}

int main() {
	string s="5",v="12";
	Karsilastir(s,v);
	return 0;
}
```

```
#include <iostream>
using namespace std;

template <typename Nesne>
void Karsilastir(Nesne &x,Nesne &y) {
	if(x>y)cout<<"Buyuk";
	else if(x<y)cout<<"Kucuk";
	else cout<<"Esit";
}

class Sayi{
	private:
		int deger;
	public:
		Sayi(int s):deger(s){}
		int Deger(){
			return deger;
		}
		bool operator>(Sayi &s)
		{
			return (deger > s.Deger());
		}
		bool operator<(Sayi &s){
			return (deger < s.Deger());
```

```
                }
};

int main() {
        Sayi *s1 = new Sayi(100);
        Sayi *s2 = new Sayi(25);

        Karsilastir(*s1,*s2);
        delete s1;
        delete s2;
        return 0;
}
```

Sınıf Şablonları

```
#include <iostream>
using namespace std;

template <typename Nesne>
class Sayi{
        private:
                Nesne deger;
        public:
                Sayi(Nesne s):deger(s){}
                Nesne Deger(){
                        return deger;
                }
};

int main(){
        Sayi<int> *s1 = new Sayi<int>(50);
        Sayi<float> *s2 = new Sayi<float>(21.84);

        cout<<s1->Deger()<<endl;
        cout<<s2->Deger();
        return 0;
}
```