

Programlama Dillerinin Prensipleri

Altprogramların İşletimi

Etkinlik Kayıtları

- Bir altprogramdaki değerler ve değişkenler için bellek ataması denetim altprograma aktarıldığında yapılır.

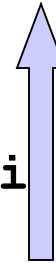
- ALGOL60 ve izleyen dillerde, altprogramlar arasında parametre iletişimi çalışma zamanındaki belleğin yığıt bölümü aracılığıyla gerçekleşir.
- Yığıt bellek, yerel değişkenler ve formal parametreler, geri dönüş adresi gibi bilgileri saklamak için kullanılır.
- (*Last In First Out, LIFO*)

- Etkin olan her altprogram için yığıt bellekte bir **etkinlik kaydı** (*activation record*) oluşturulur. Altprogramlarda kullanılan yerel değişkenler için her yordama ilişkin etkinlik kaydı kapsamında, yığıt bellekte bellek atanır.

C++ Fonksiyon Örneği

Top of Stack

Yığıtın büyümesi



```
void sub(double total, int part)
{
    int list[5];
    double sum;
    ...
}
```

Local	sum
Local	list [4]
Local	list [3]
Local	list [2]
Local	list [1]
Local	list [0]
Parameter	part
Parameter	total
Dynamic link	
Static link	
Return address	

Aktivasyon Kaydı

```

void fun1(int x) {
    int y;
    ...
    fun3(y);
    ...
}
void fun2(double r) {
    int s,t;
    ...
    fun1(s);
    ...
}
void fun3(int q) {
    ...
}
void main()
{
    double p;
    ...
    fun2(p);
    ...
}

```

Aktivasyon Kaydı(main)	Yerel Değişken	p
---------------------------	-------------------	---

← Top





```
void fun1(int x) {
    int y;
    ...
    fun3(y) ;
    ...
}
void fun2(double r) {
    int s,t;
    ...
    fun1(s) ;
    ...
}
void fun3(int q) {
    ...
}
void main()
{
    double p;
    ...
    fun2 (p) ;
    ...
}
```



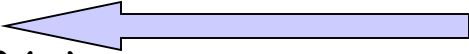
	YİĞİT TEPEŚİ	
	Yerel deęiřkenler	t
	Yerel deęiřkenler	s
Fun2 iin aktivasyon kaydı	Formal parametre	r
	Dinmik baę (link)	
	Main'e dnüş	
Main iin aktivasyon kaydı	Yerel deęiřkenler	p



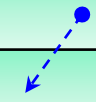
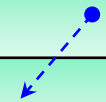
```

void fun1(int x) {
    int y;
    ...
    fun3(y);
    ...
}
void fun2(double r) {
    int s,t;
    ...
    fun1(s);
    ...
}
void fun3(int q) {
    ...
}
void main()
{
    double p;
    ...
    fun2(p);
    ...
}

```



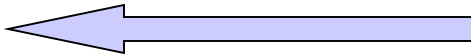
	YIĞITIN TEPEŞİ	
	Yerel değişken	y
Fun1 için aktivasyon kaydı	Formal parametre	x
	Dinamik link	
	Fun2'ye dönüş	
	Yerel değişkenler	t
	Yerel değişkenler	s
Fun2 için aktivasyon kaydı	Formal parametre	r
	Dinamik bağ (link)	
	Main'e dönüş	
Main için aktivasyon kaydı	Yerel değişkenler	p




```

void fun1(int x) {
    int y;
    ...
    fun3(y);
    ...
}
void fun2(double r) {
    int s,t;
    ...
    fun1(s);
    ...
}
void fun3(int q) {
    ...
}
void main()
{
    double p;
    ...
    fun2(p);
    ...
}

```

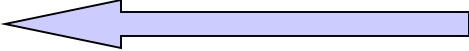


Fun3 için aktivasyon kaydı	YİĞİTİN TEPEŞİ	
	Formal parametre	q
	Dinamik link	
	Fun1'e dönüş	
Fun1 için aktivasyon kaydı	Yerel değişken	y
	Formal parametre	x
	Dinamik link	
	Fun2'ye dönüş	
Fun2 için aktivasyon kaydı	Yerel değişkenler	t
	Yerel değişkenler	s
	Formal parametre	r
	Dinamik bağ (link)	
	Main'e dönüş	
Main için aktivasyon kaydı	Yerel değişkenler	p

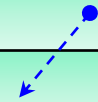
```

void fun1(int x) {
    int y;
    ...
    fun3(y);
    ...
}
void fun2(double r) {
    int s,t;
    ...
    fun1(s);
    ...
}
void fun3(int q) {
    ...
}
void main()
{
    double p;
    ...
    fun2(p);
    ...
}

```



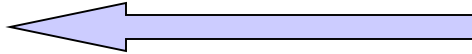
	YİĞİTİN TEPEŞİ	
	Yerel değişken	y
Fun1 için aktivasyon kaydı	Formal parametre	x
	Dinamik link	
	Fun2'ye dönüş	
	Yerel değişkenler	t
	Yerel değişkenler	s
Fun2 için aktivasyon kaydı	Formal parametre	r
	Dinamik bağ (link)	
	Main'e dönüş	
Main için aktivasyon kaydı	Yerel değişkenler	p



```

void fun1(int x) {
    int y;
    ...
    fun3(y);
    ...
}
void fun2(double r) {
    int s,t;
    ...
    fun1(s);
    ...
}
void fun3(int q) {
    ...
}
void main()
{
    double p;
    ...
    fun2(p);
    ...
}

```



	YIĞITIN TEPESİ	
	Yerel değişkenler	t
	Yerel değişkenler	s
Fun2 için aktivasyon kaydı	Formal parametre	r
	Dinamik bağ (link)	
	Main'e dönüş	
Main için aktivasyon kaydı	Yerel değişkenler	p



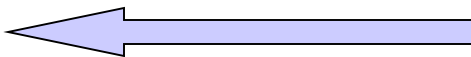
```

void fun1(int x) {
    int y;
    ...
    fun3(y);
    ...
}
void fun2(double r) {
    int s,t;
    ...
    fun1(s);
    ...
}
void fun3(int q) {
    ...
}
void main()
{
    double p;
    ...
    fun2(p);
    ...
}

```

Top
↙

	YIĞITIN TEPEŚİ	
Main için aktivasyon kaydı	Yerel deęişkenler	p



```


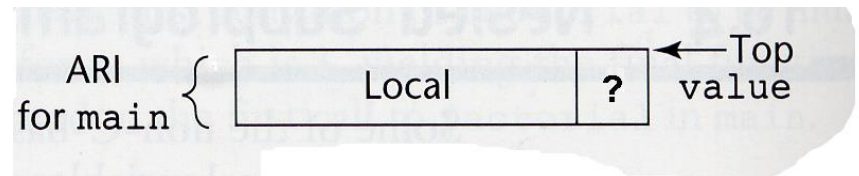
void factorial(int x) {
    ...
    if (n <= 1)
    {
        return 1;
    }
    else
    {
        return (n*factorial(n-1));
    }
}

```

```

void main()
{
    int value;
    ...
    value = factorial(3);
    ...
}

```

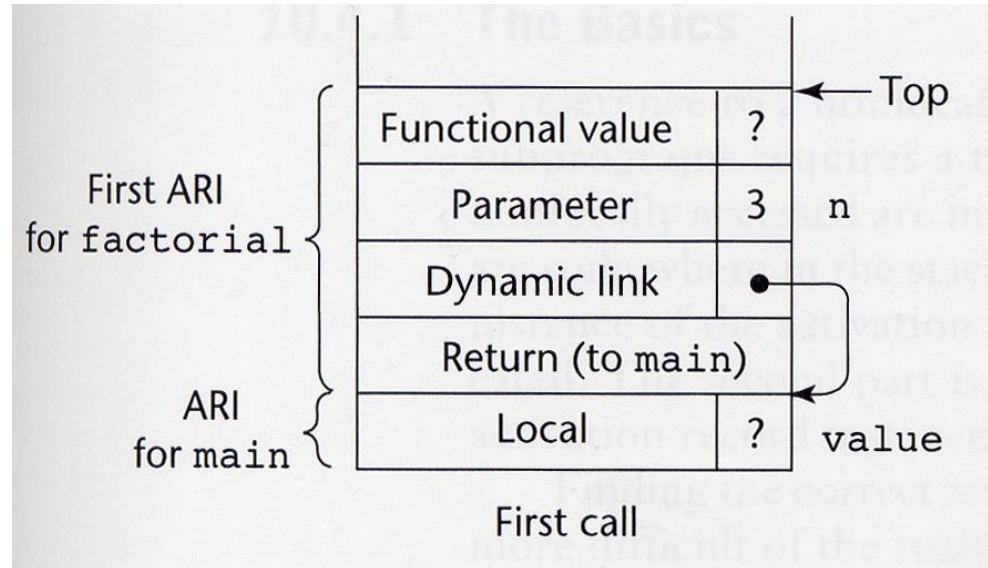



```

void factorial(int x) {
    ...
    if (n <= 1)
    {
        return 1;
    }
    else
    {
        return (n*factorial(n-1));
    }
}

void main()
{
    int value;
    ...
    value = factorial(3);
    ...
}

```

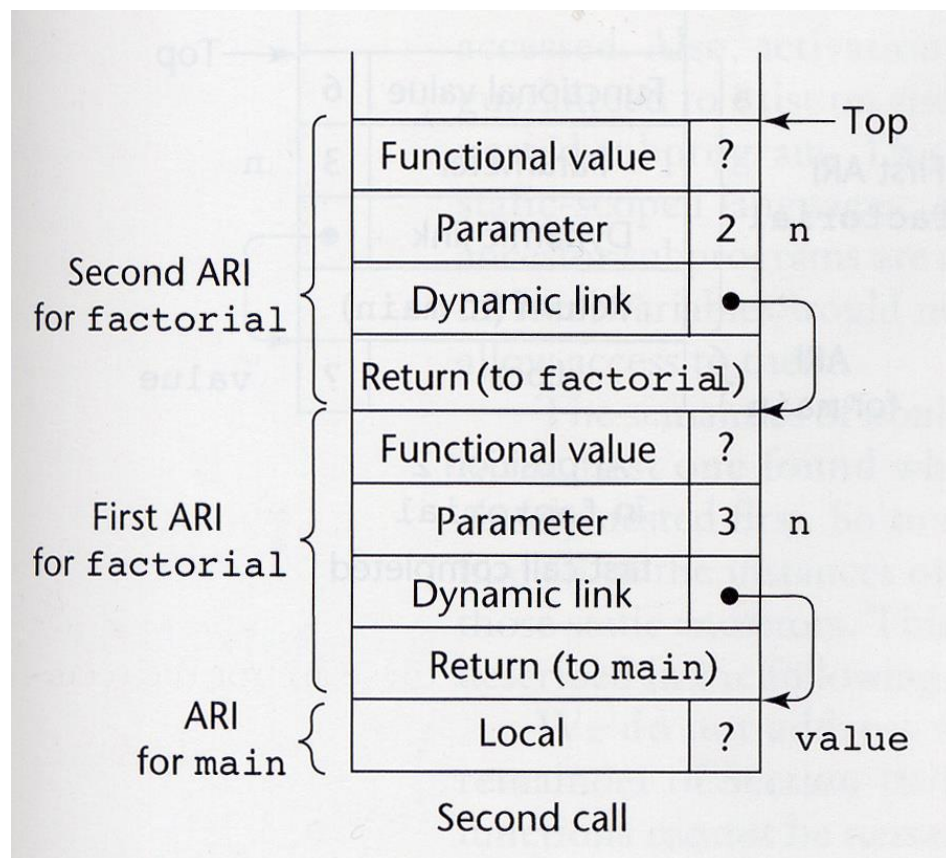


```

void factorial(int x) {
    ...
    if (n <= 1)
    {
        return 1;
    }
    else
    {
        return (n*factorial(n-1));
    }
}

void main()
{
    int value;
    ...
    value = factorial(3);
    ...
}

```

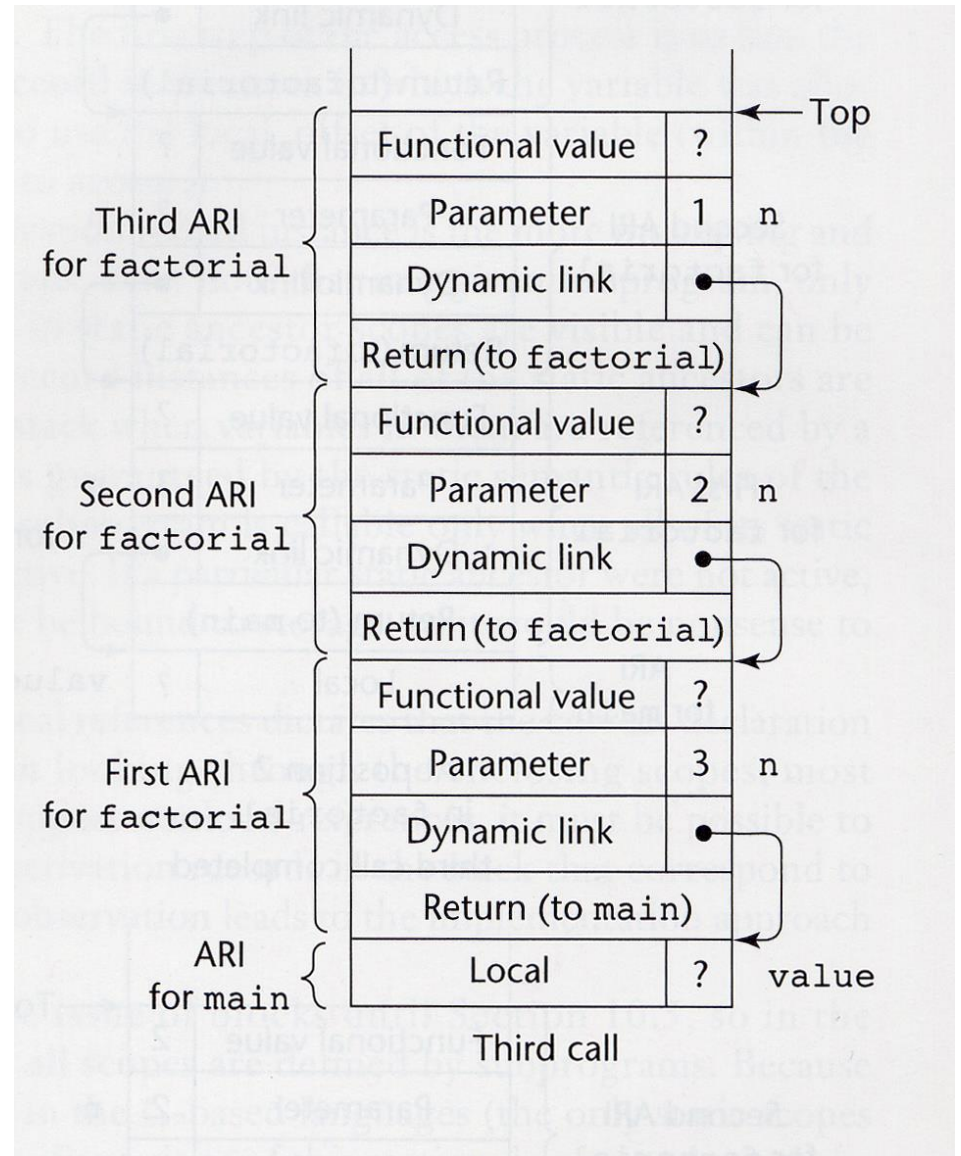


```

void factorial(int x) {
    ...
    if (n <= 1)
    {
        return 1;
    }
    else
    {
        return (n*factorial(n-1));
    }
}

void main()
{
    int value;
    ...
    value = factorial(3);
    ...
}

```

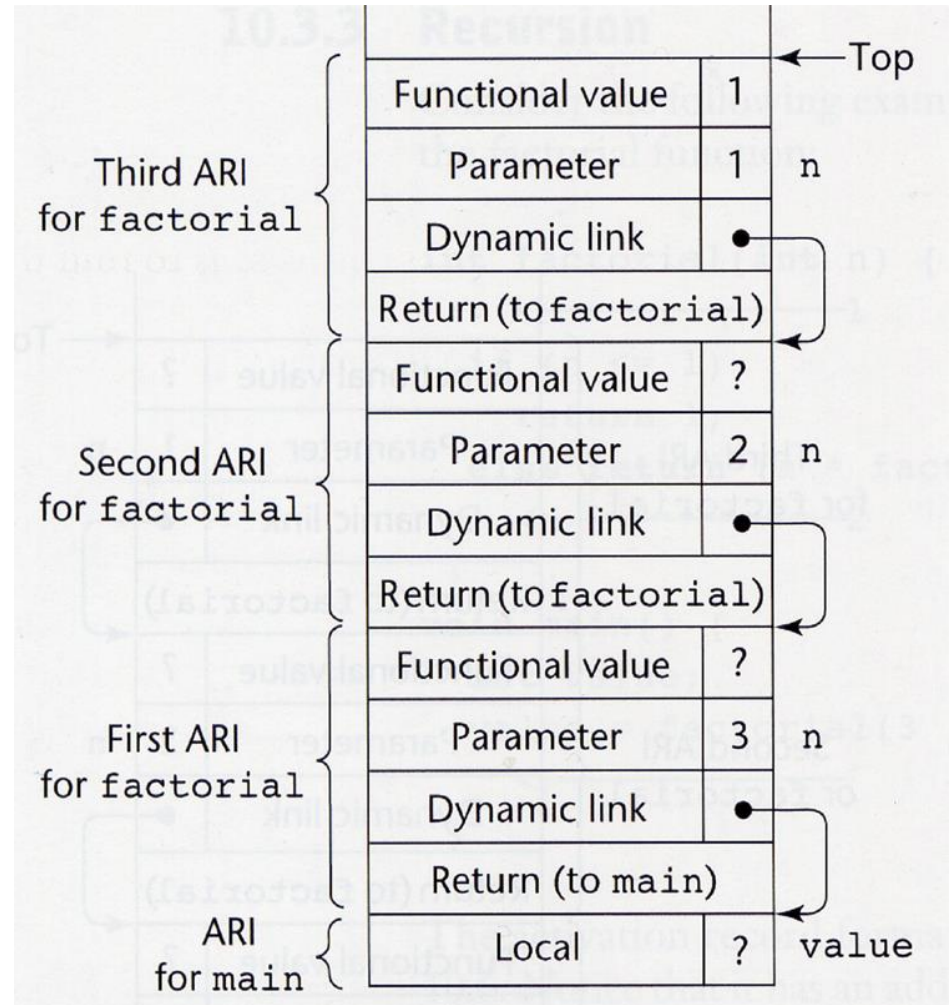



```

void factorial(int x) {
    ...
    if (n <= 1)
    {
        return 1;
    }
    else
    {
        return (n*factorial(n-1));
    }
}

void main()
{
    int value;
    ...
    value = factorial(3);
    ...
}

```

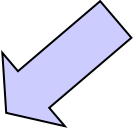
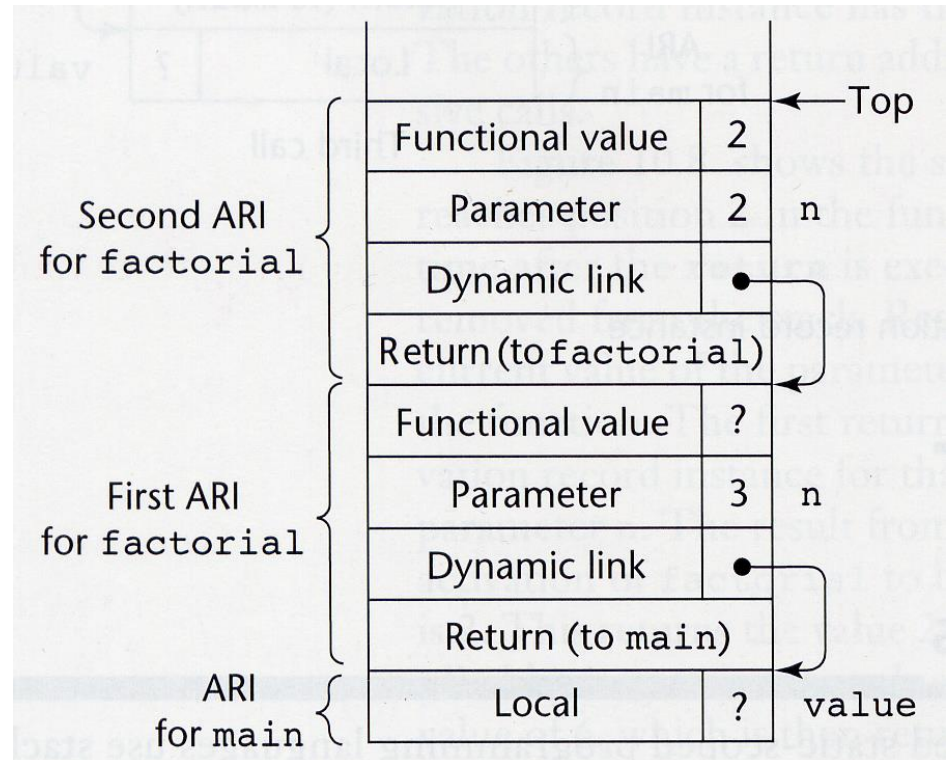


```

void factorial(int x) {
    ...
    if (n <= 1)
    {
        return 1;
    }
    else
    {
        return (n*factorial(n-1));
    }
}

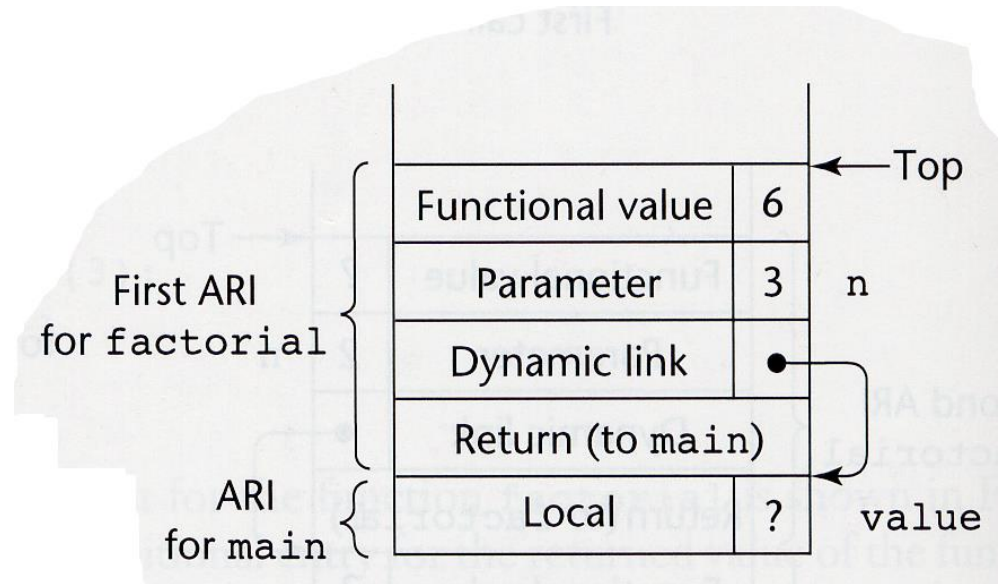
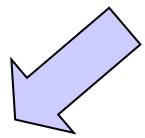
void main()
{
    int value;
    ...
    value = factorial(3);
    ...
}

```

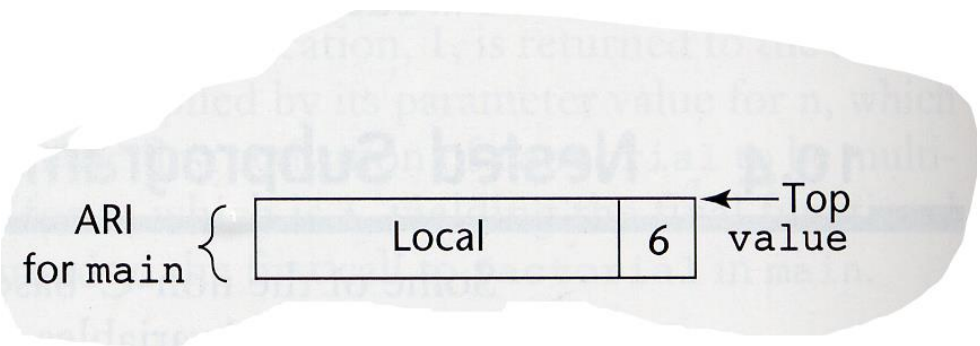
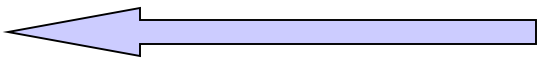
```
void factorial(int x) {
    ...
    if (n <= 1)
    {
        return 1;
    }
    else
    {
        return (n*factorial(n-1));
    }
}

void main()
{
    int value;
    ...
    value = factorial(3);
    ...
}
```



```
void factorial(int x) {
    ...
    if (n <= 1)
    {
        return 1;
    }
    else
    {
        return (n*factorial(n-1));
    }
}
```

```
void main()
{
    int value;
    ...
    value = factorial(3);
    ...
}
```



```
program MAIN_2;
  var X : integer;
  procedure BIGSUB;
    var A, B, C : integer;

    procedure SUB1;
      var A, D : integer;
      begin { SUB1 }
        A := B + C;  <-----1
      end; { SUB1 }
    procedure SUB2(X : integer);
      var B, E : integer;

      procedure SUB3;
        var C, E : integer;
        begin { SUB3 }
          SUB1;
          E := B + A; <-----2
        end; { SUB3 }

      begin { SUB2 }
        SUB3;
        A := D + E;  <-----3
      end; { SUB2 }

    begin { BIGSUB }
      SUB2(7);
    end; { BIGSUB }

begin
  BIGSUB;
end. { MAIN_2 }
```

MAIN_2 calls BIGSUB
BIGSUB calls SUB2
SUB2 calls SUB3
SUB3 calls SUB1

	YığıtınTepesi	
Aktivasyon kaydı (SUB1)	yerel	D
	Yerel	A
	Dinamik link	
	Static link	
	Return (sub3)	
Aktivasyon kaydı (SUB3)	Yerel	E
	Yerel	C
	Dinamik link	
	Static link	
	Return (sub2)	
Aktivasyon kaydı (SUB2)	Yerel	E
	Yerel	B
	Parametre	7, X
	Dinamik link	
	Static link	
	Return (BIGSUB)	
Aktivasyon kaydı (BIGSUB)	Yerel	C
	Yerel	B
	Yerel	A
	Dinamik link	
	Static link	
Aktivasyon kaydı (main_2)	Return (main_2)	
	Yerel Değişken	x

Dynamic Kapsam

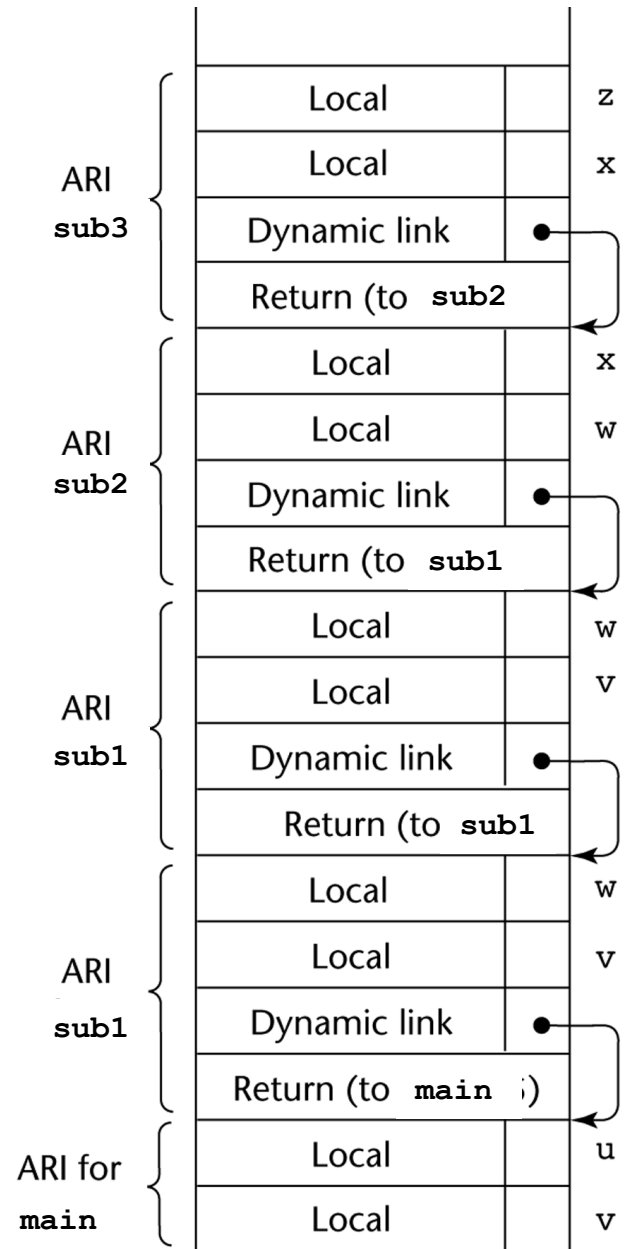
```
void sub3()
{
    int x, z;
    x = u + v;
    ...
}

void sub2()
{
    int w,x;
    ...
}

void sub1()
{
    int v, w;
    ...
}

void main()
{
    int v,u;
    ...
}
```

```
main sub1'i
sub1 sub1'i
sub1 sub2'yi
sub2 sub3'ü
çağırırsın
```



ARI = activation record instance