Recursion Function

1. A function which is called by itself is called recursion function.

2. In recursive function, whenever a function is called stack memory is used internally for storing the function return addresses.

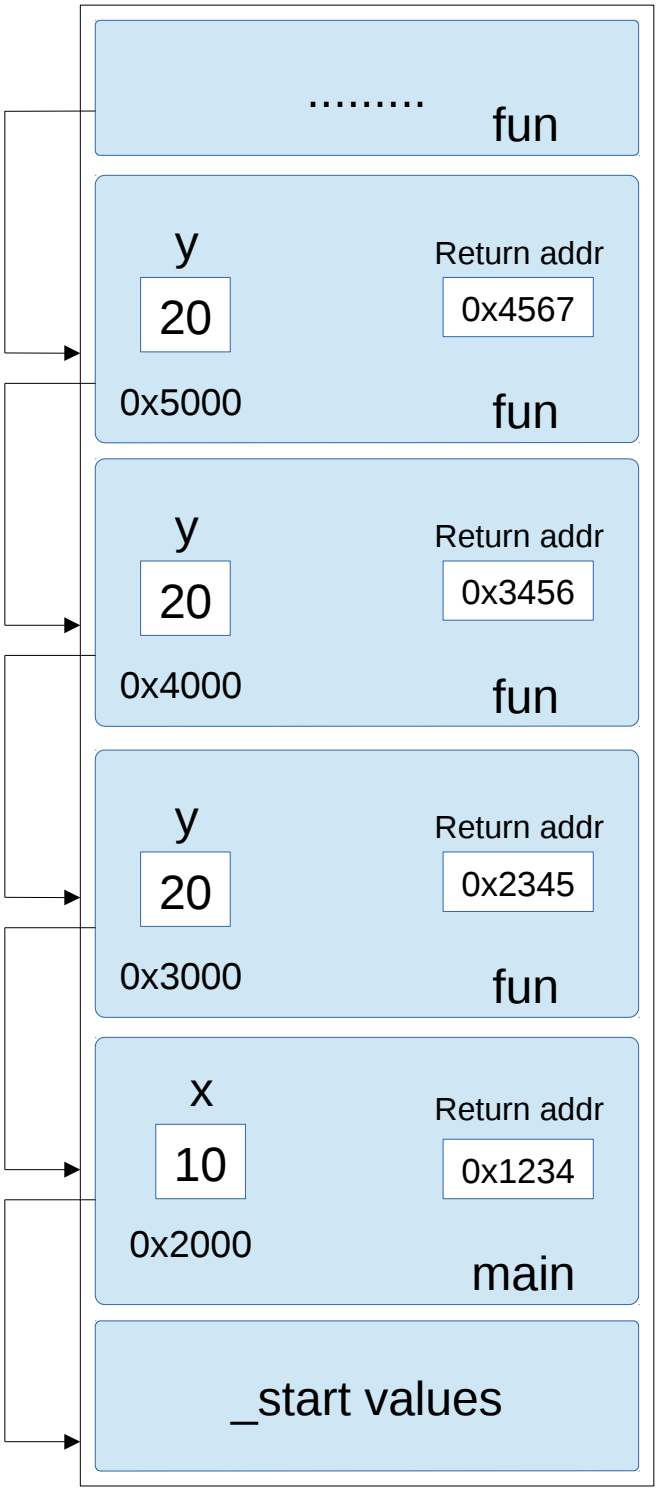
3. If the recursion is happening continuously, so that once stack memory is completely filled with function return addresses, then it leads to segmentation fault.

Advantage : It reduces code complexity

Disadvantages :

1. Takes more memory in stack.
2. makes slower in execution because of many push & pop operations

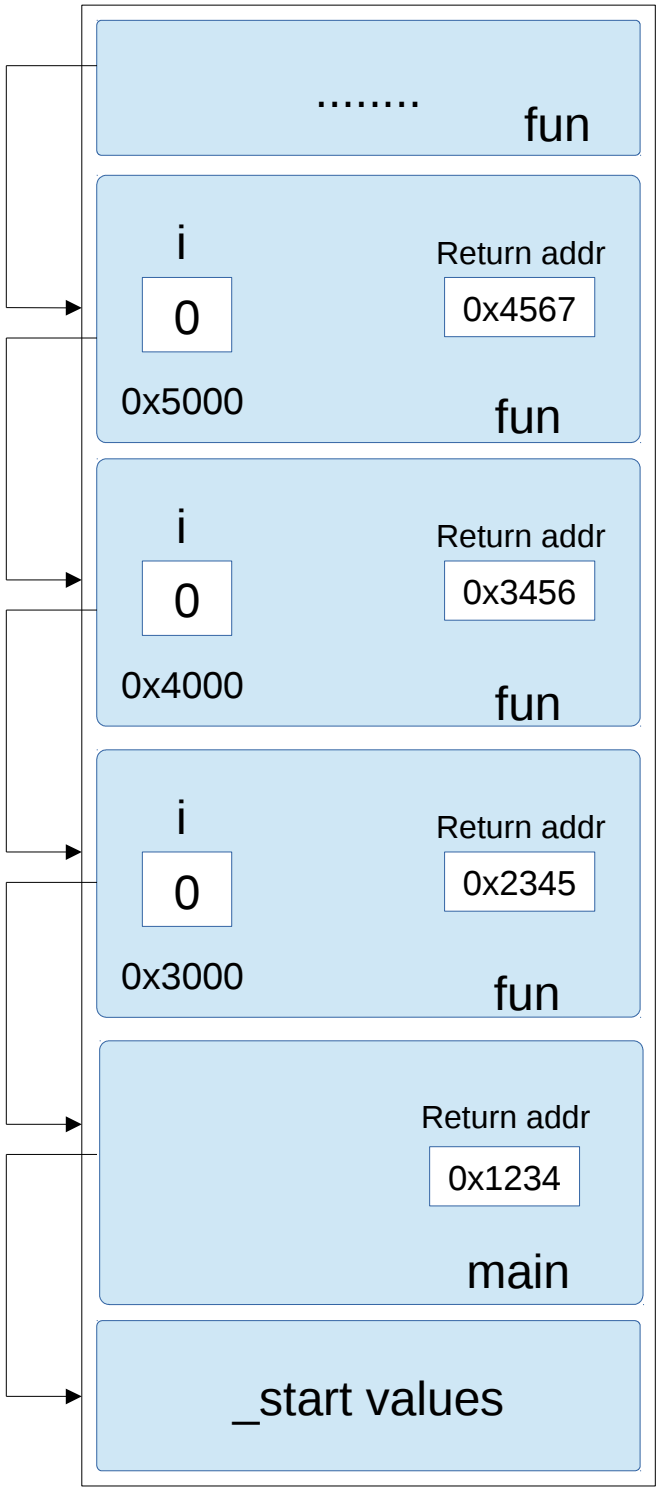
```
1 #include<stdio.h>
2 void fun();
3 int main()
4 {
5         int x = 10;
6         printf("before fun()\n");
7         fun();
8         printf("after fun()...\n");
9         printf("x = %d\n",x);
10 }
11 void fun()
12 {
13         int y = 20;
14         printf("In fun(), y = %d\n",y);
15         fun();
16 }
```



Stack

```
void fun()
{
   int y = 20;
   printf("In fun(), y = %d\n",y);
   fun();
}

void fun()
{
   int y = 20;
   printf("In fun(), y = %d\n",y);
   fun();
}

void fun()
{
   int y = 20;
   printf("In fun(), y = %d\n",y);
   fun();
}

int main()
{
  int x = 10;
  printf("before fun()\n");
  fun();
  printf("after fun()...\n");
  printf("x = %d\n",x);
}
```

```
1 #include<stdio.h>
2 void fun();
3 int main()
4 {
5         printf("In main()...\n");
6         fun();
7 }
8 void fun()
9 {
10        int i = 0;
11        if(i++ < 3) {
12        printf("In fun(), i = %d\n",i);
13        fun();
14        }
15 }
```

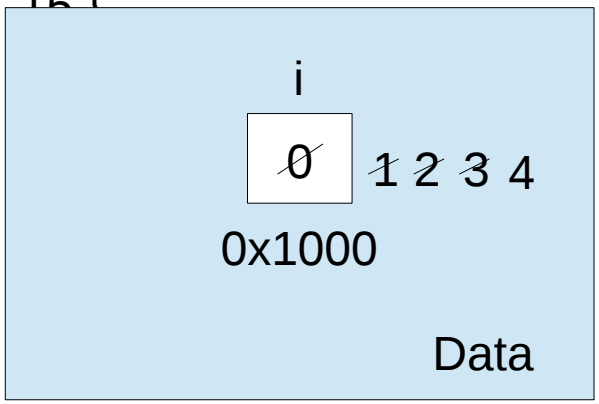


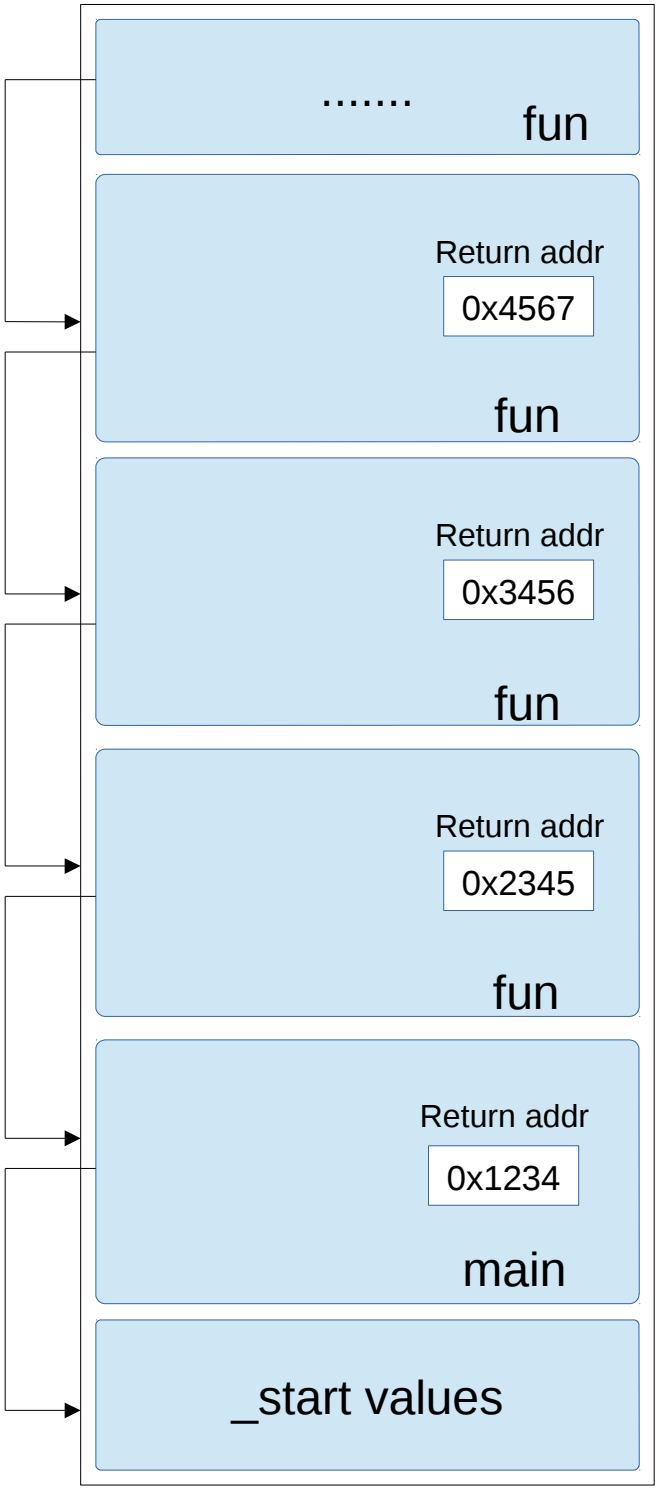
Stack

3

```
void fun()
{
   int i = 0;
   if(i++ < 3) {
   printf("In fun(), i = %d\n",i);
   fun();
   }
}void fun()
{
   int i = 0;
   if(i++ < 3) {
   printf("In fun(), i = %d\n",i);
   fun();
   }
}void fun()
{
   int i = 0;
   if(i++ < 3) {
   printf("In fun(), i = %d\n",i);
   fun();
   }
}

int main()
{
   printf("In main()...\n");
   fun();
}
```

```
1 #include<stdio.h>
2 void fun();
3 int main()
4 {
5         printf("In main()...\n");
6         fun();
7 }
8 void fun()
9 {
10        static int i = 0;
11        if(i++ < 3) {
12        printf("In fun(), i =
%d\n",i);
13        fun();
14        }
15 }
```



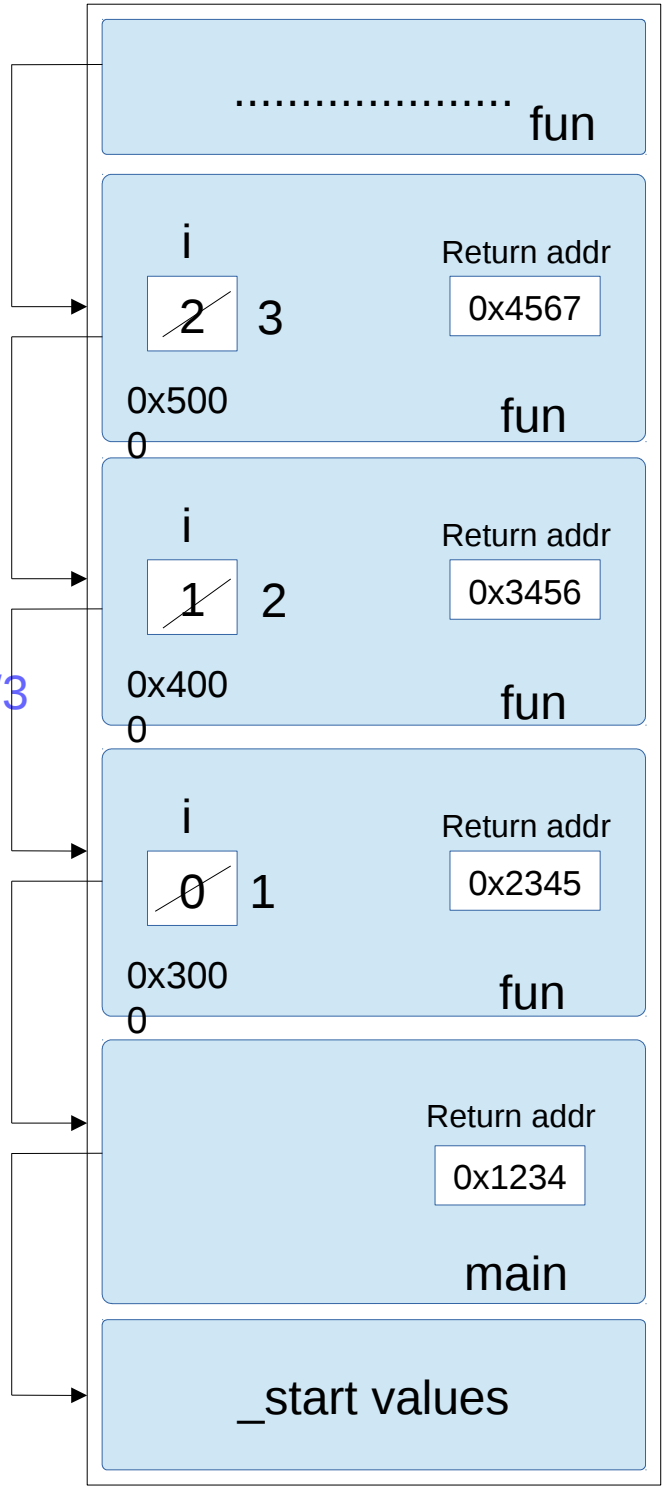


Stack

4

```
void fun()
{
   int i = 0;
   if(i++ < 3) {
   printf("In fun(),i = %d\n",i)
   fun();
   }
}
void fun()
{
   int i = 0;
   if(i++ < 3) {
   printf("In fun(),i = %d\n",i)
   fun();
   }
}
void fun()
{
   int i = 0;
   if(i++ < 3) {
   printf("In fun(),i = %d\n",i)
   fun();
   }
}

int main()
{
   printf("In main()...\n");
   fun();
 }
```

```c
1 #include<stdio.h>
2 void fun(int);
3 int main()
4 {
5         printf("In main()...\n");
6         fun(0);
7 }
8 void fun(int i)//0 //1 //2 //3
9 {
10        if(i++ < 3) //0<3 //1<3 //2<3
11        {
12                printf("In fun(), i = %d\n",i);//1//2//3
13                fun(i); //fun(1); //fun(2); //fun(3)
14        }
15 }
```

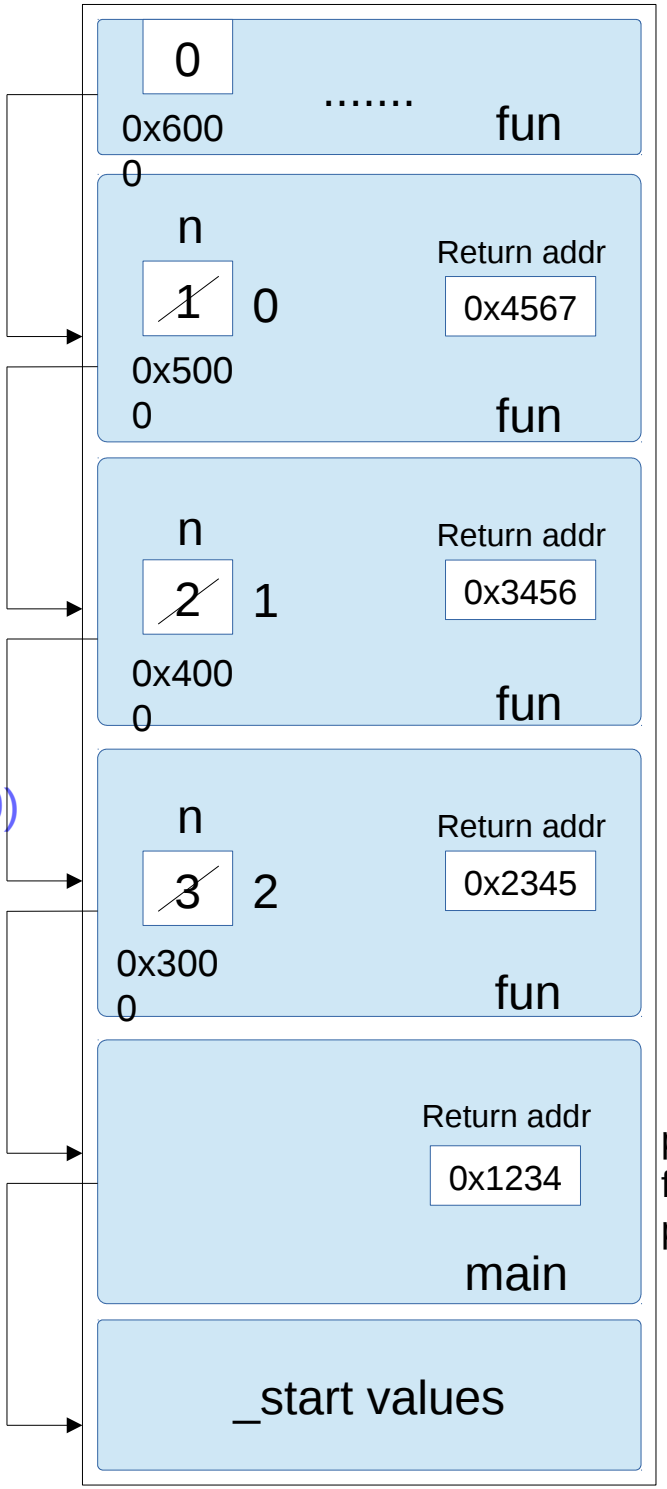


Stack

5

```c
void fun(int i)
{
   if(i++ < 3) {
   printf("In fun(), i = %d\n",i);
   fun();
   }
}
void fun(int i)
{
   if(i++ < 3) {
   printf("In fun(), i = %d\n",i);
   fun();
   }
}
void fun(int i)
{
   if(i++ < 3) {
   printf("In fun(), i = %d\n",i);
   fun();
   }
}

int main()
{
   printf("In main()...\n");
   fun();
}
```

```
1 #include<stdio.h>
2 void fun(int n);
3 int main()
4 {
5         printf("In main(),before fun()\n");
6         fun(3);
7         printf("\nIn main(),after fun()\n");
8 }
9 void fun(int n) //3 //2 //1
10 {
11         if(n>0)
12         {
13                 printf("%d  ",n); //3 //2 //1
14                 fun(--n); //fun(2) //fun(1) //fun(0)
15         }
16
17         printf("\nHello...");
18 }
```



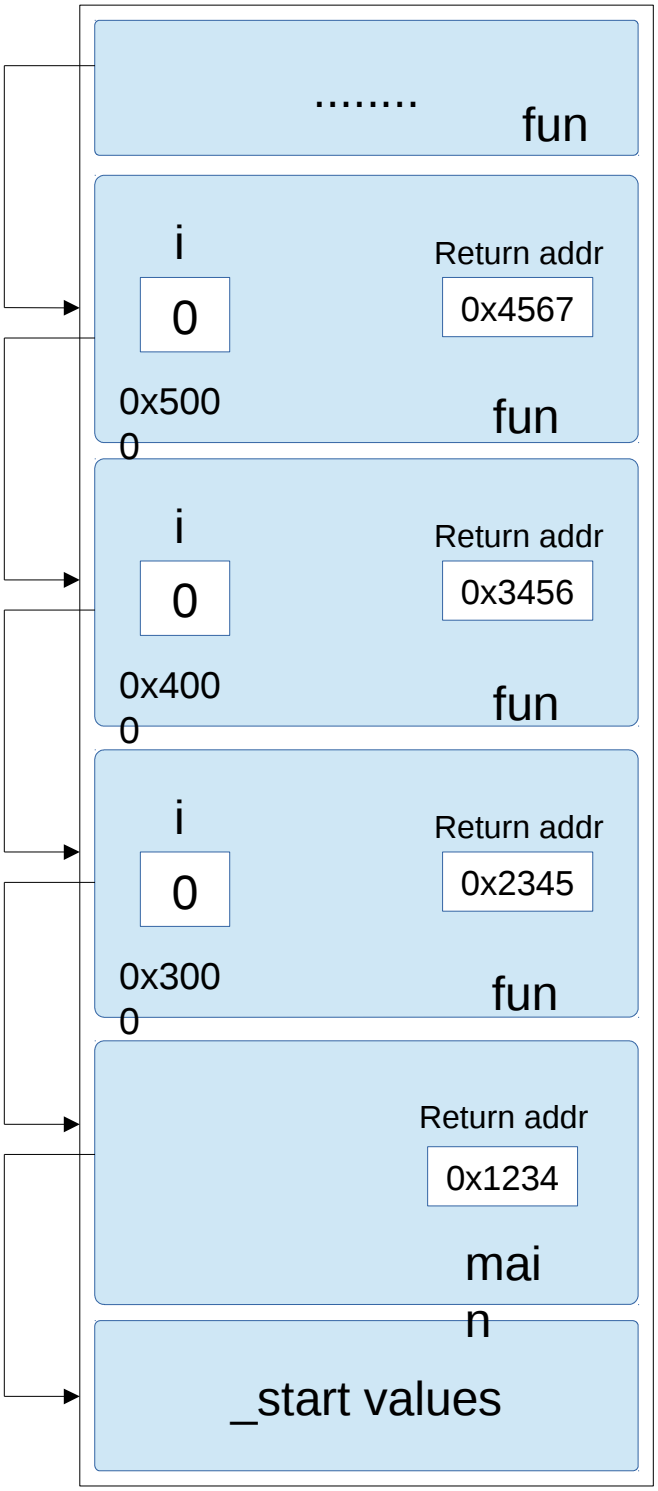
Stack

6

```
if(n>0)
{
   pf("%d  ",n);
   fun(--n);
}
pf("\nHello...");

if(n>0)
{
   pf("%d  ",n);
   fun(--n);
}
pf("\nHello...");

if(n>0)
{
   pf("%d  ",n);
   fun(--n);
}
pf("\nHello...");
```

```
pf("In main(),before fun()\n");
fun(3);
pf("\nIn main(),after fun()\n");
```

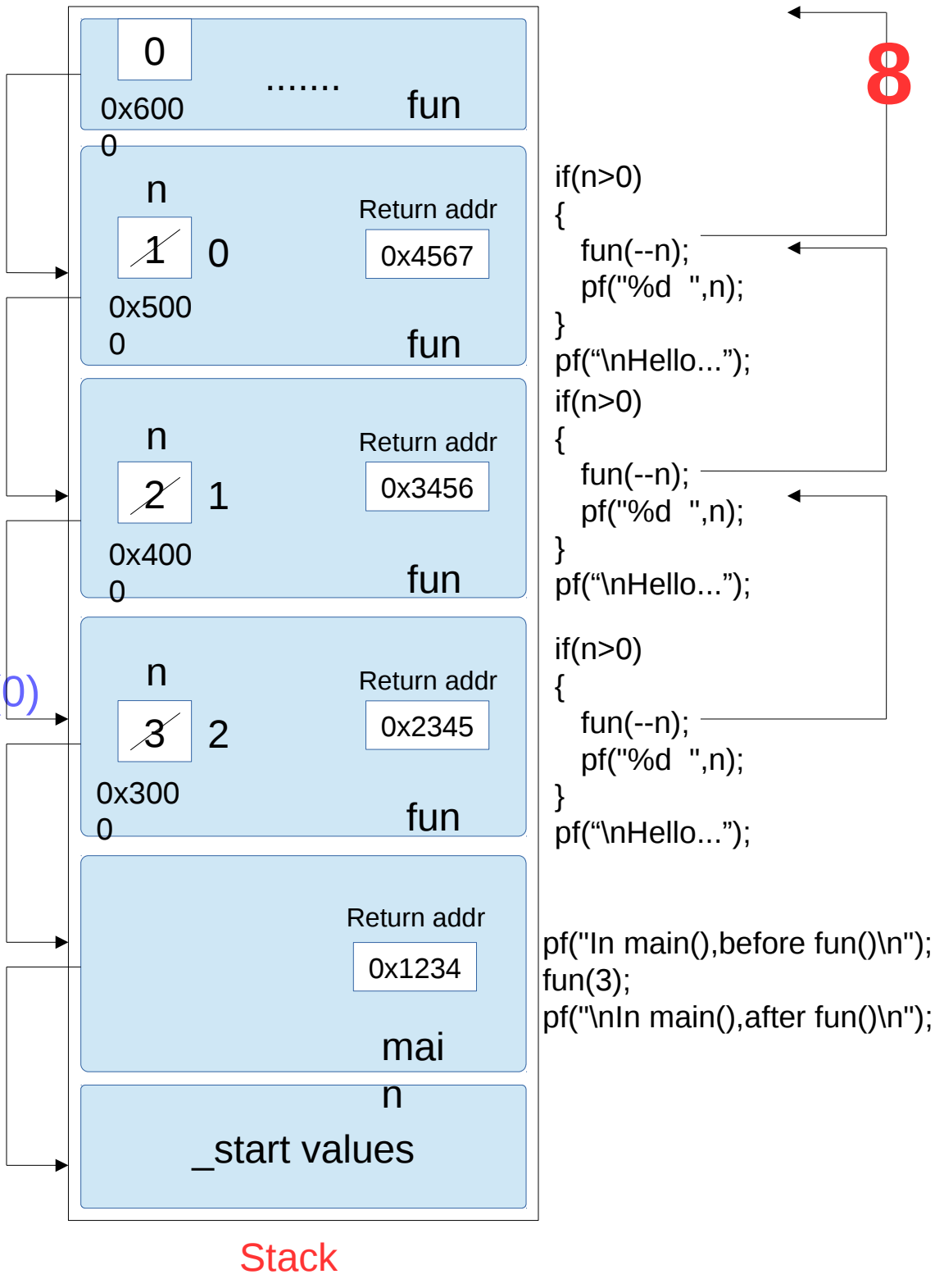
```
1 #include<stdio.h>
2 void fun();
3 int main()
4 {
5         printf("In main()...\n");
6         fun();
7 }
8 void fun()
9 {
10        int i = 0;
11        if(i++ < 3) {
12        printf("In fun(), i =%d\n",i);
13        fun();
14        }
15 }
```



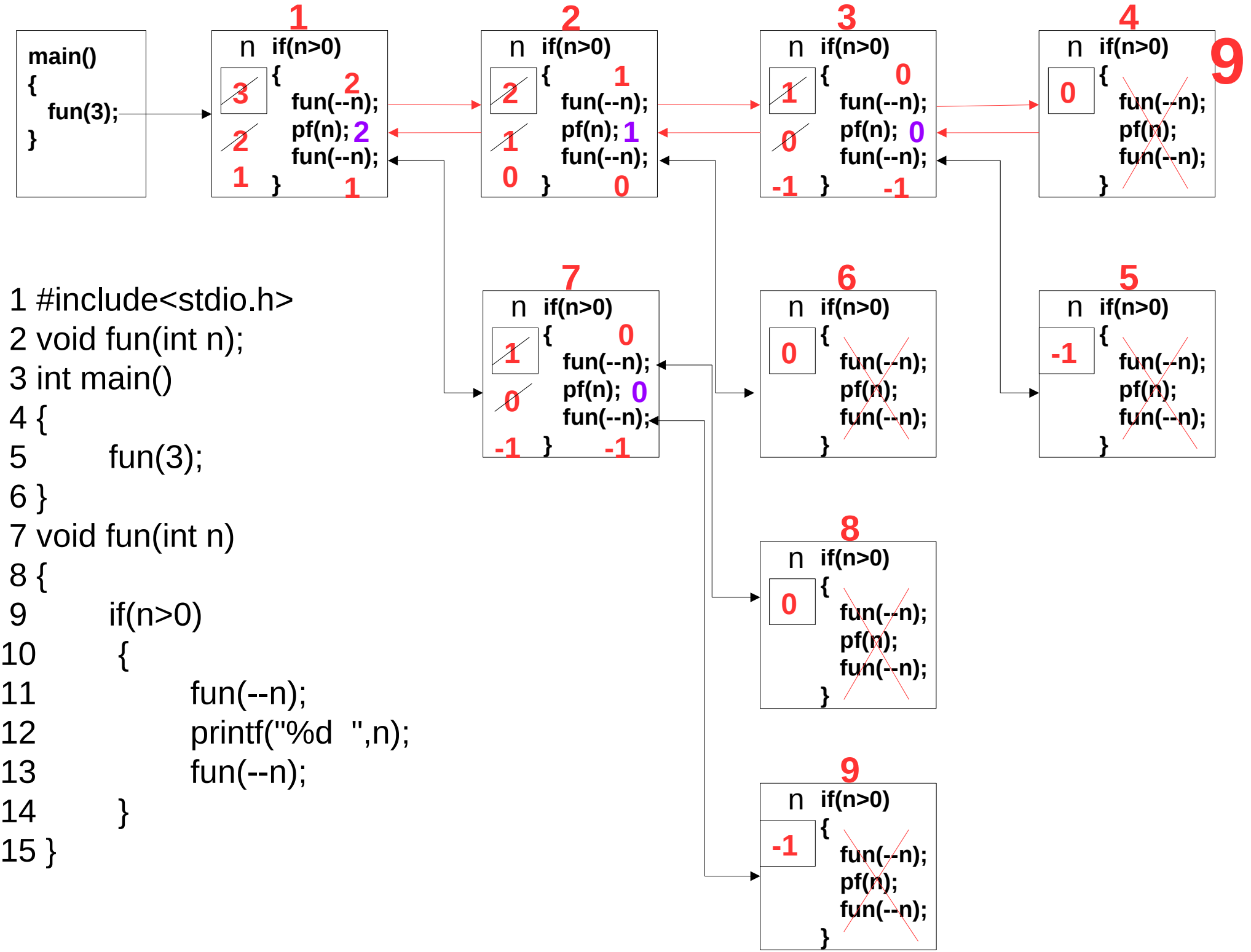

```
void fun()
{
   int i = 0;
   if(i++ < 3) {
   printf("In fun(), i = %d\n",i);
   fun();
   }
}
void fun()
{
   int i = 0;
   if(i++ < 3) {
   printf("In fun(), i = %d\n",i);
   fun();
   }
}
void fun()
{
   int i = 0;
   if(i++ < 3) {
   printf("In fun(), i = %d\n",i);
   fun();
   }
}

int main()
{
   printf("In main()...\n");
   fun();
}
```

Stack

```
1 #include<stdio.h>
2 void fun(int n);
3 int main()
4 {
5         printf("In main(),before fun()\n");
6         fun(3);
7         printf("\nIn main(),after fun()\n");
8 }
9 void fun(int n) //3 //2 //1
10 {
11        if(n>0)
12        {
13            fun(--n); //fun(2) //fun(1) //fun(0)
14            printf("%d  ",n); //0 //1 //2
15         }
16
17         printf("\nHello...");
18 }
```

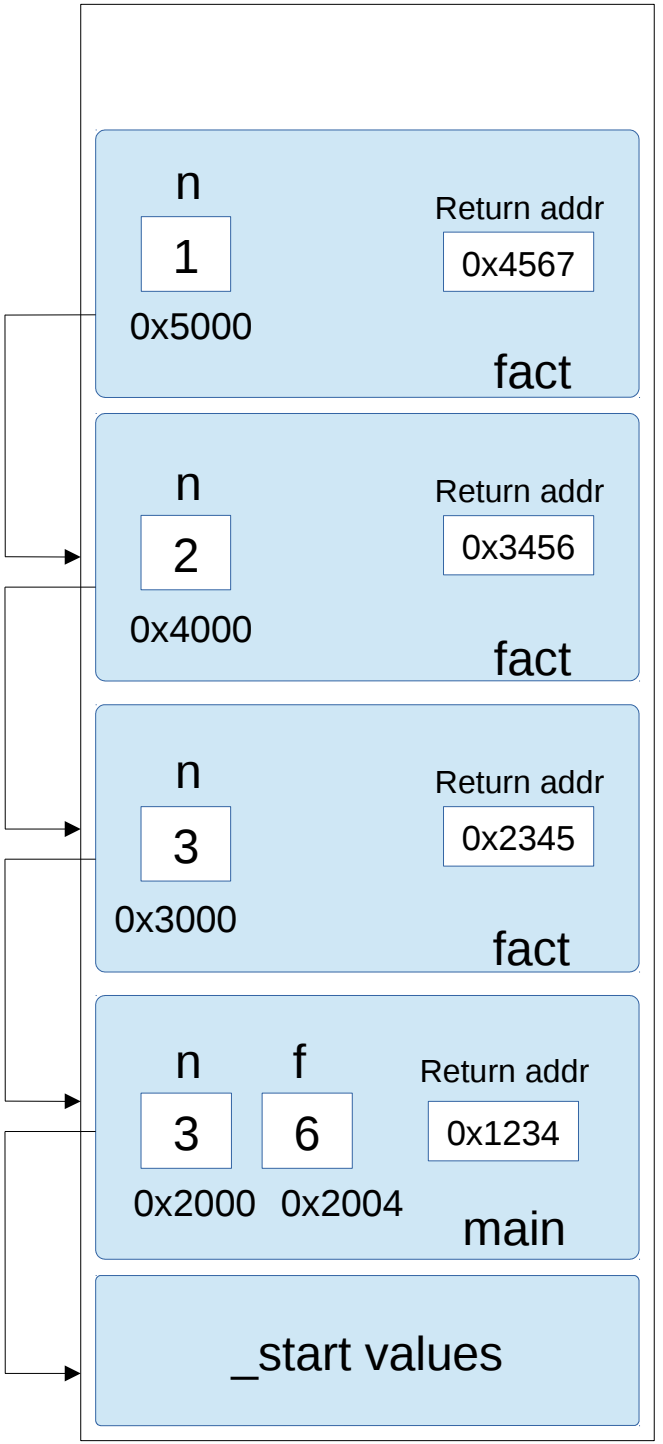


Stack




```
1 #include<stdio.h>
2 void fun(int n);
3 int main()
4 {
5         fun(3);
6 }
7 void fun(int n)
8 {
9         if(n>0)
10         {
11                 fun(--n);
12                 printf("%d  ",n);
13                 fun(--n);
14         }
15 }
```

```
1 #include<stdio.h>
2 int fact(int);
3 int main()
4 {
5         int n,f;
6         printf("Enter the n value\n");
7         scanf("%d",&n);
8
9         f = fact(n);
10         printf("f = %d\n",f);
11 }
12 int fact(int n)
13 {
14         if(n==1)
15         return 1;
16         else
17         return n*fact(n-1);
18 }
```



Stack

return 1;

return 2*fact(1);

1

2

return 3*fact(2);

2

6

f = fact(3);

6

//write a program to find factorial of a given number using recursion function.

```
#include<stdio.h>
int fact(int,int);
int main()
{
        int n,f;
        printf("Enter the n value\n");
        scanf("%d",&n);

        f = fact(n,1);
         printf("f = %d\n",f);
}
int fact(int n,int f)
{
         if(n!=0) //4!=0,3!=0,2!=0,1!=0,0!=0
         {
                 f = f*n; //f=1*4,f=4*3,f=12*2,f=24*1
                 n = n-1; //n=3,n=2,n=1,n=0
                 return fact(n,f);
         }
         else
         return f;

}
```

//write a progrm to find the sum of digits using recursion fun.

```
 1 #include<stdio.h>
 2 int sum(int n,int s);
 3 int main()
 4 {
 5         int n;
 6         printf("Enter the n value\n");
 7         scanf("%d",&n);
 8
 9         int s = sum(n,0);
10          printf("s = %d\n",s);
11 }
12 int sum(int n,int s)
13 {
14          if(n!=0)
15          {
16                  s = s+n%10;
17                  n = n/10;
18                  return sum(n,s);
19          }
20          else
21          return s;
22 }
```

//write a program to reverse the string using recurion function.

```
1 #include<stdio.h>
2 #include<string.h>
3 void rev_str(char *,int,int);
4 int main()
5 {
6         char s[50];
7         printf("Enter the string\n");
8         scanf("%s",s);
9
10         rev_str(s,0,strlen(s)-1);
11         printf("s = %s\n",s);
12 }
13 void rev_str(char *p,int i,int j)
14 {
15         char temp;
16         if(i<j)
17         {
18                 temp = p[i];
19                 p[i] = p[j];
20                 p[j] = temp;
21                 i++, j--;
22                 rev_str(p,i,j);
23         }
24 }
```