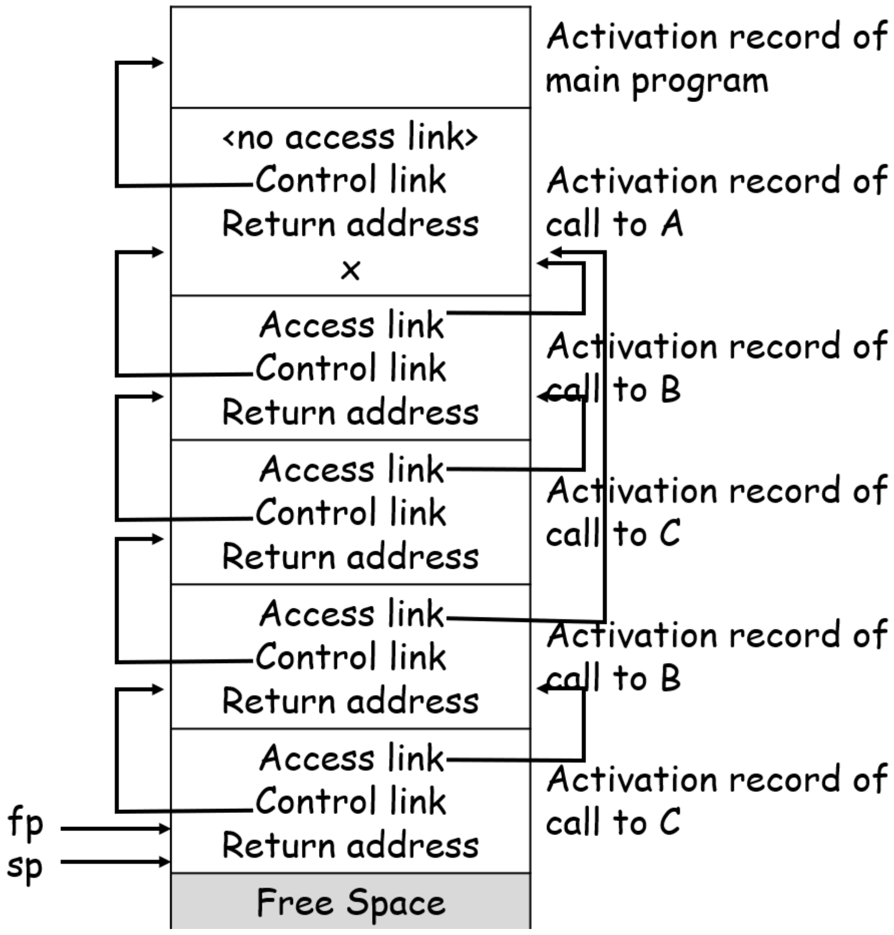
Compile Principle - HW of Chapter 7

解雲暄 3190105871

> 7.4 Draw the stack of activation records for the following Pascal program, showing the control and access links, after the second call to procedure `c`. Describe how the variable `x` is accessed from within `c`.
>
> ```pascal
> program env;
>
> procedure a;
> varx: integer;
>     procedure b;
>         procedure c;
>         begin
>             x := 2;
>             b;
>         end;
>     begin (* b *)
>         c;
>     end;
> begin (* a *)
>     b;
> end;
>
> begin (* main *)
>     a;
> end;
> ```



7.15 Give the output of the following program (written in C syntax) using the 4 parameter passing methods discussed in Section 7.5:

```c
#include <stdio.h>
int i = 0;

void p(int x, int y)
{
    x += 1;
    i += 1;
    y += 1;
}

main()
{
    int a[2]={1,1};
    p(a[i], a[i]);
    printf("%d %d\n",a[0],a[1]);
    return 0;
17  }
```

Pass by value: 1 1

`i = 0`, and `p(1, 1)` is called, but `a[0]` and `a[1]` are not modified.

Pass by reference: 3 1

`i = 0`, so `p(a[0], a[0])` is called. After `x += 1` and `y += 1`, `a[0]` becomes 3. `a[1]` is never accessed.

Pass by value-result: 2 1

`i = 0`, so `p(a[0], a[0])` is called, `x = 1`, `y = 1`. After `x += 1` and `y += 1`, `x = 2`, `y = 2`. We now put `x` into `a[0]` and `a[0] = 2`; then we put `y` into `a[0]` so `a[0] = 2`. `a[1]` is never accessed.

Pass by name: 2 2

We expand Line 14 to `a[i] += 1; i += 1; a[i] += 1;`, which actually does: `a[0] = 2`, `i = 2`, `a[1] = 2`.