Question: How are function calls "implemented" behind the scenes?

CPU basically executes instructions in the sequence they appear in memory.
Current instruction's address stored in the "instruction pointer" or "program counter" or "IP", "PC".

How to handle function calls?

"Runtime stack" (or "call stack").

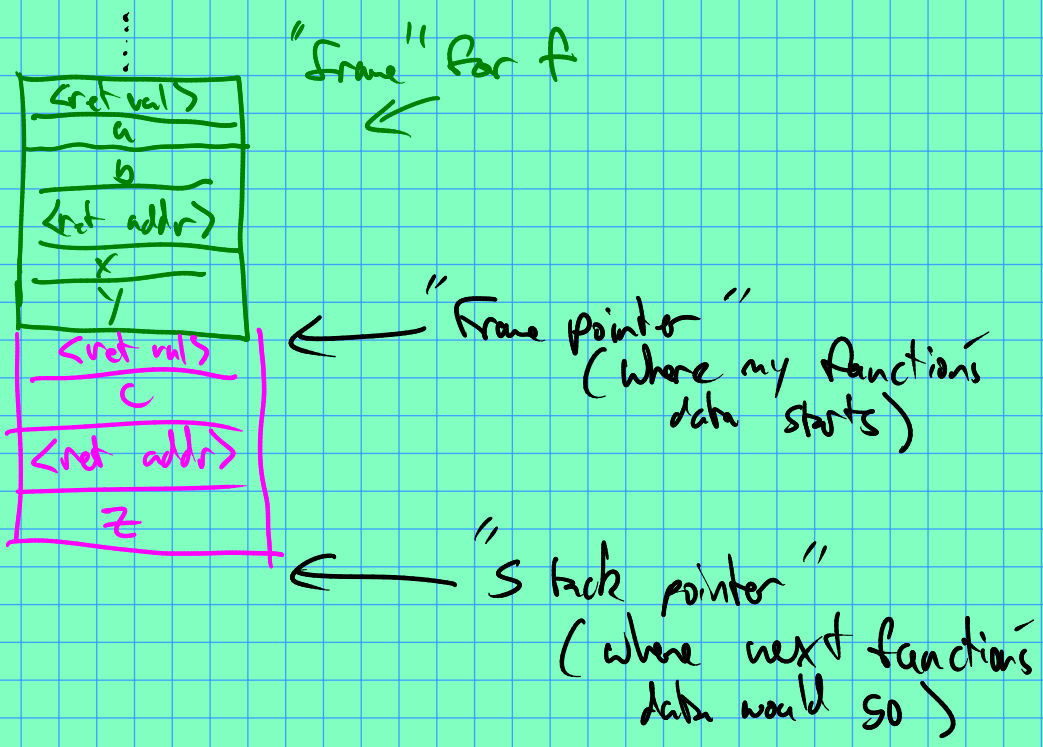
- Keeps track of functions currently running.
- Stores the following information for each call:
  - Storage for parameters.
  - Storage for local variables.
  - "return address" (i.e. pointer for "code after the call")
  - Storage for return value.

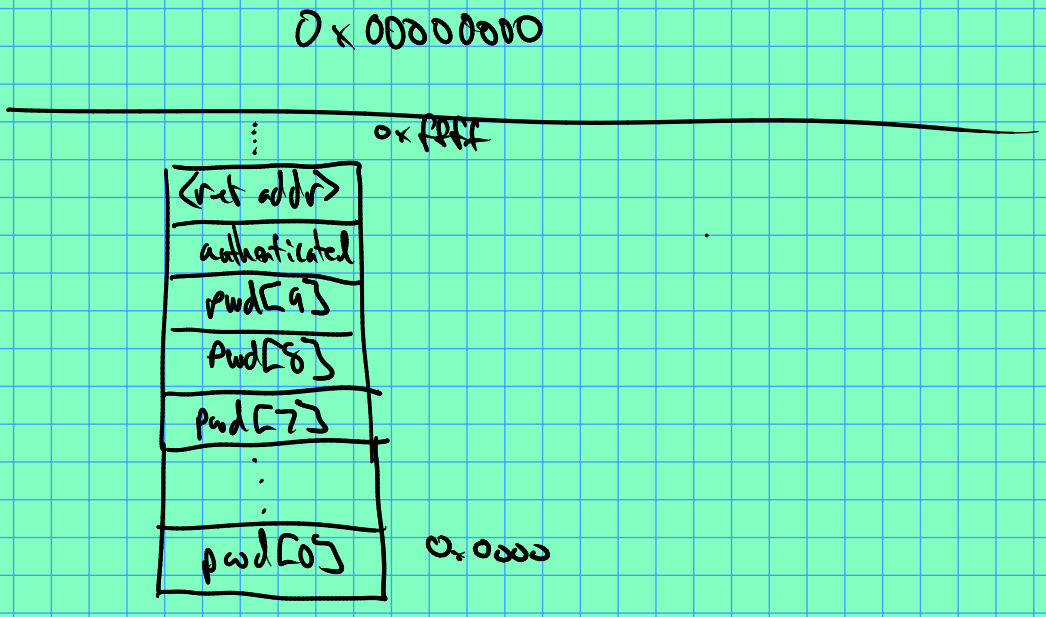
E.x.:

```
int f(int a, int b)
{   int x, y;
    y = g(x);
    ...
    return 0;
}
```

```
int g(int c)
{   int z;
    ...
    return 42;
}
```

0xffffffff


"frame" for f
<ret val>
a
b
<ret addr>
x
y
"Frame pointer"
(where my functions data starts)
<ret val>
c
<ret addr>
z
"Stack pointer"
(where next functions data would go)


0x00000000
0xFFFF
<ret addr>
authenticated
pwd[9]
pwd[8]
pwd[7]
pwd[0]
0x0000