Unit 2 Embedded Programs and Tools

Models of Program

Embedded code must not only provide rich functionality, it must also often run at a required rate to meet system deadlines, fit into the allowed amount of memory, and meet power consumption requirements..

Designing code that simultaneously meets multiple design constraints is a considerable challenge, but luckily there are techniques and tools that we can use to help us through the design process.

We can also build a CDFG for an assembly language program.

Aug 28-11:39

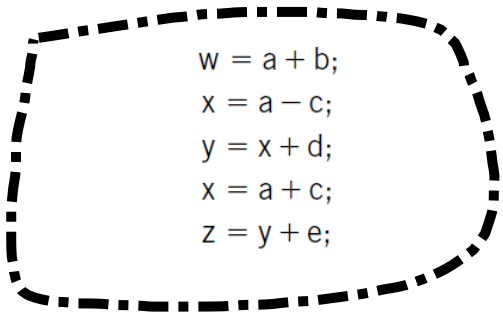


FIGURE 5.2

A basic block in C.

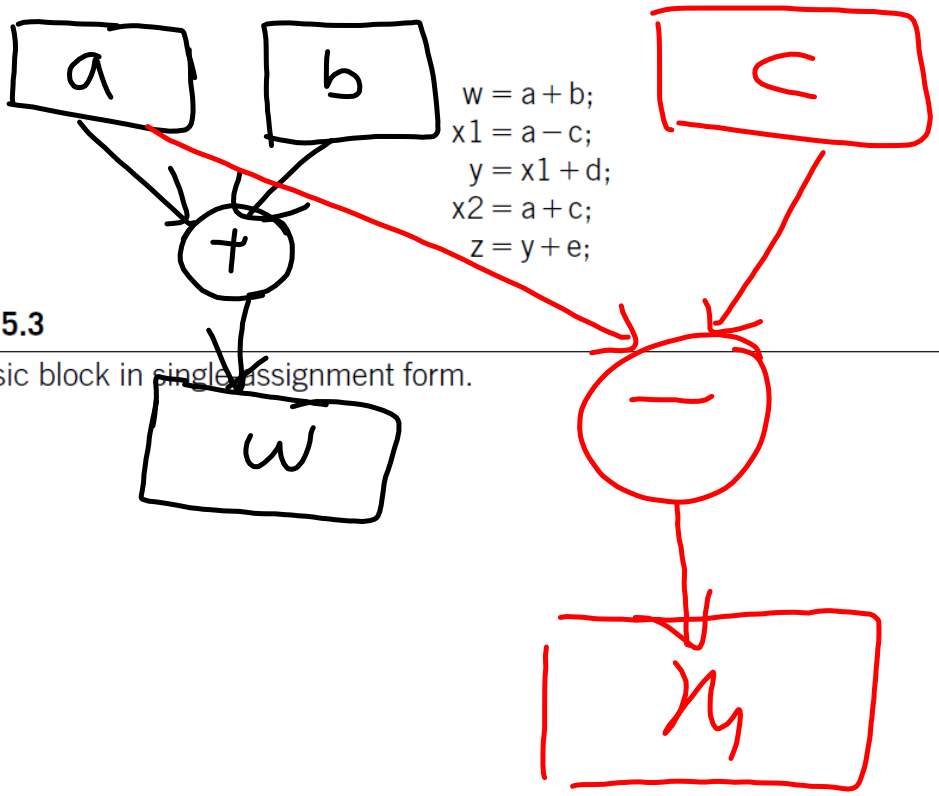
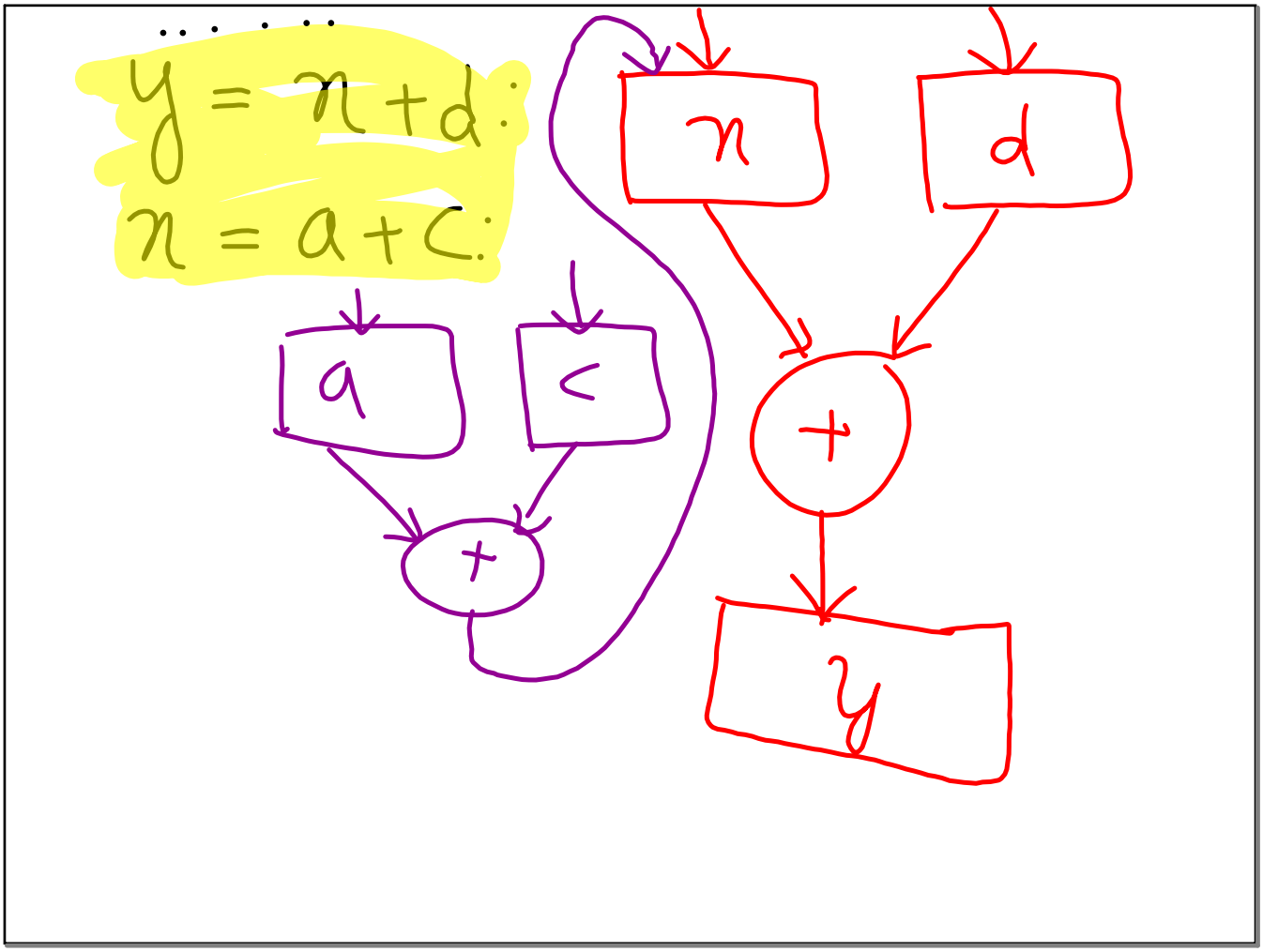
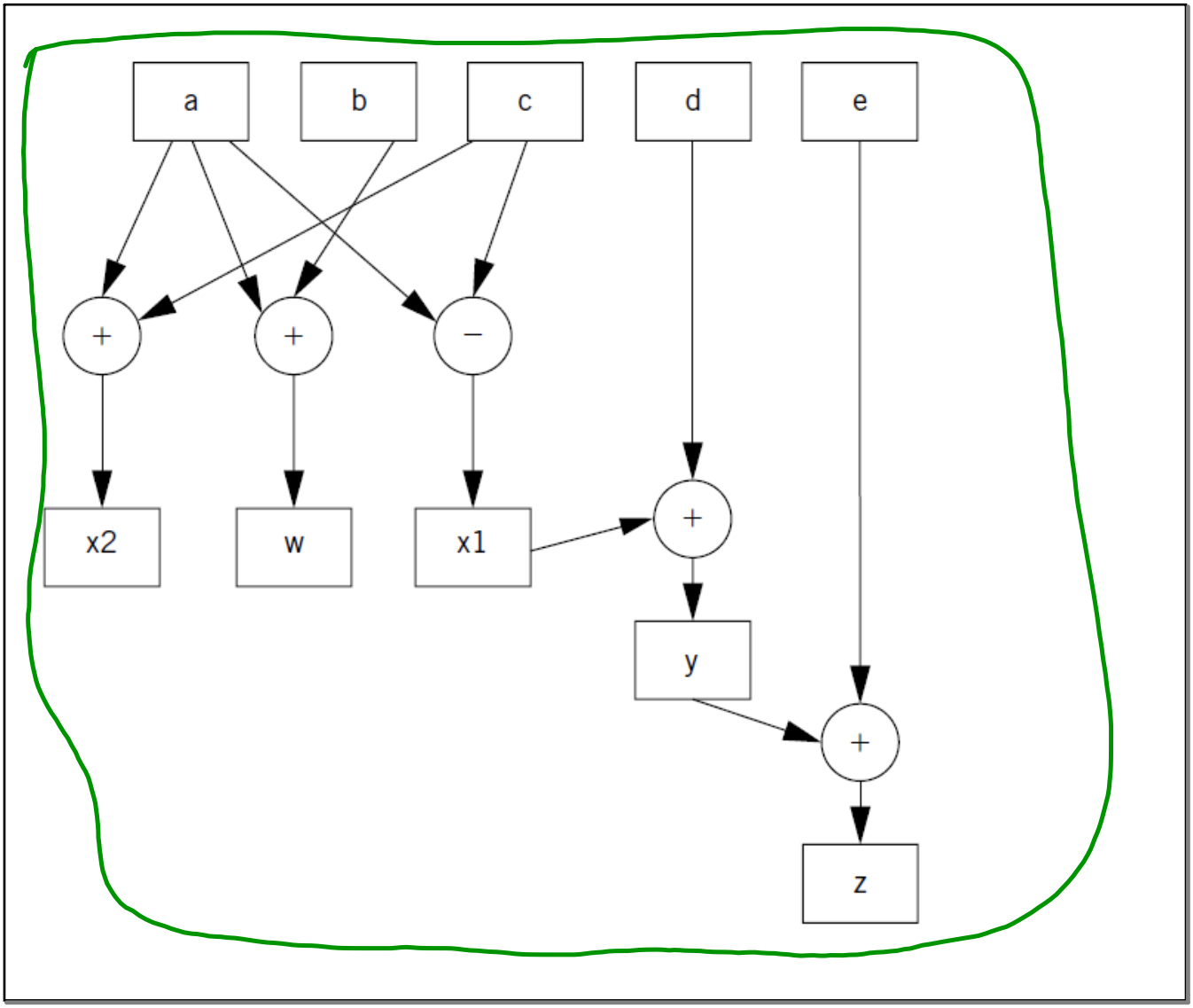


FIGURE 5.3

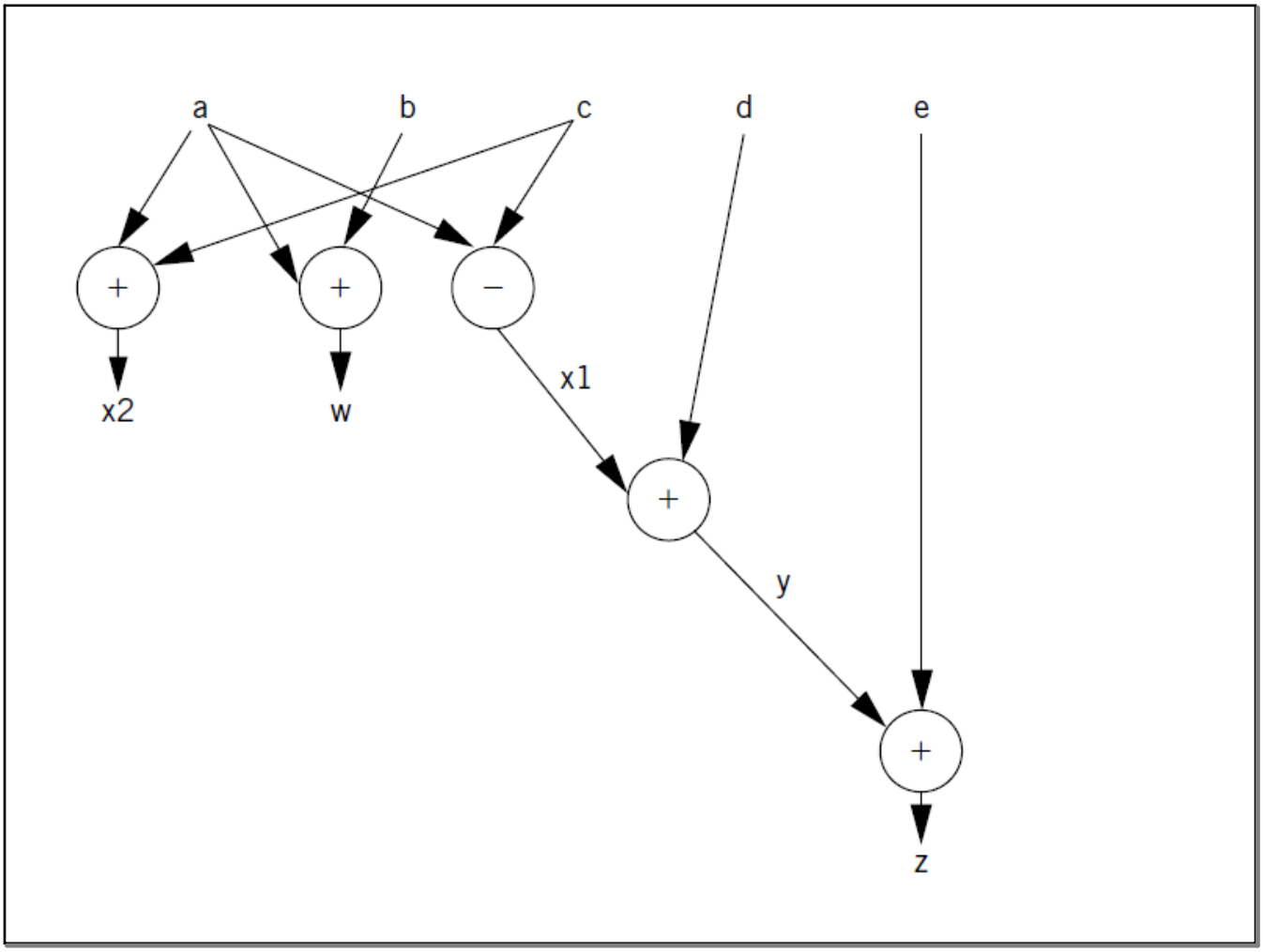
The basic block in single-assignment form.



Aug 31-16:21



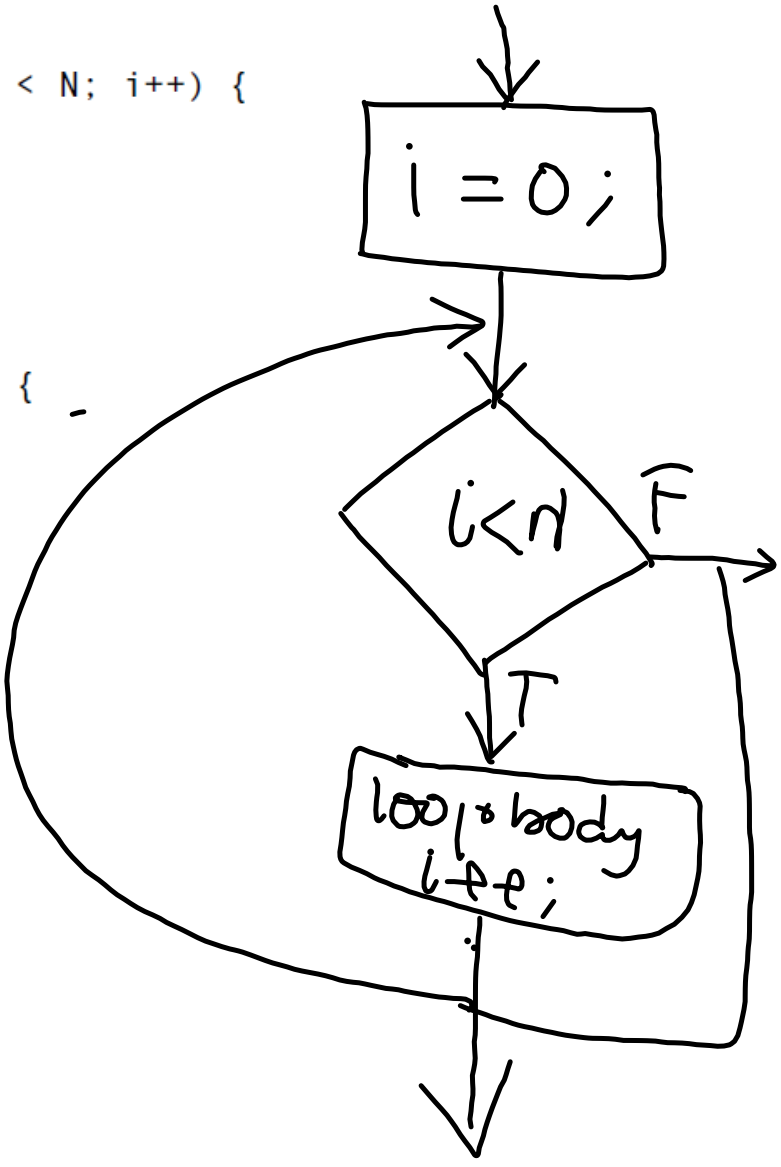
Aug 29-08:55


a
b
c
d
e
+
+
−
x2
w
x1
+
y
+
z

```
for (i = 0; i < N; i++) {
loop_body();
}
```

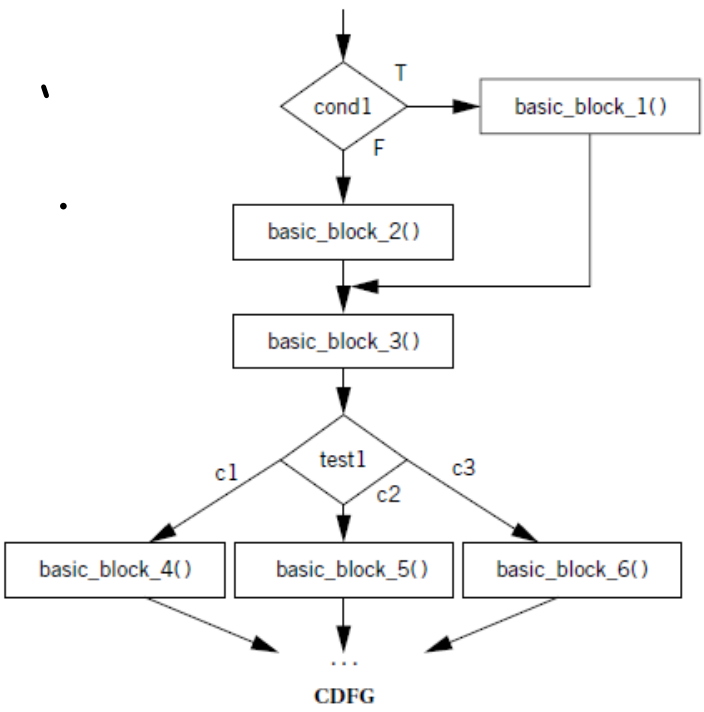
is equivalent to

```
i = 0;
while (i < N) {
loop_body();
i++;
}
```



```
if (cond1)
      basic_block_1();
else
      basic_block_2();
basic_block_3();
switch (test1) {
      case c1: basic_block_4(); break;
      case c2: basic_block_5(); break;
      case c3: basic_block_6(); break;
}
```

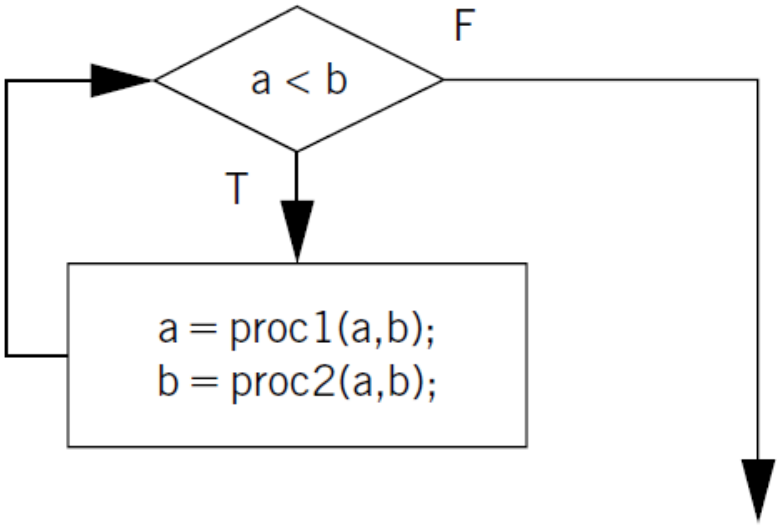
C code



CDFG

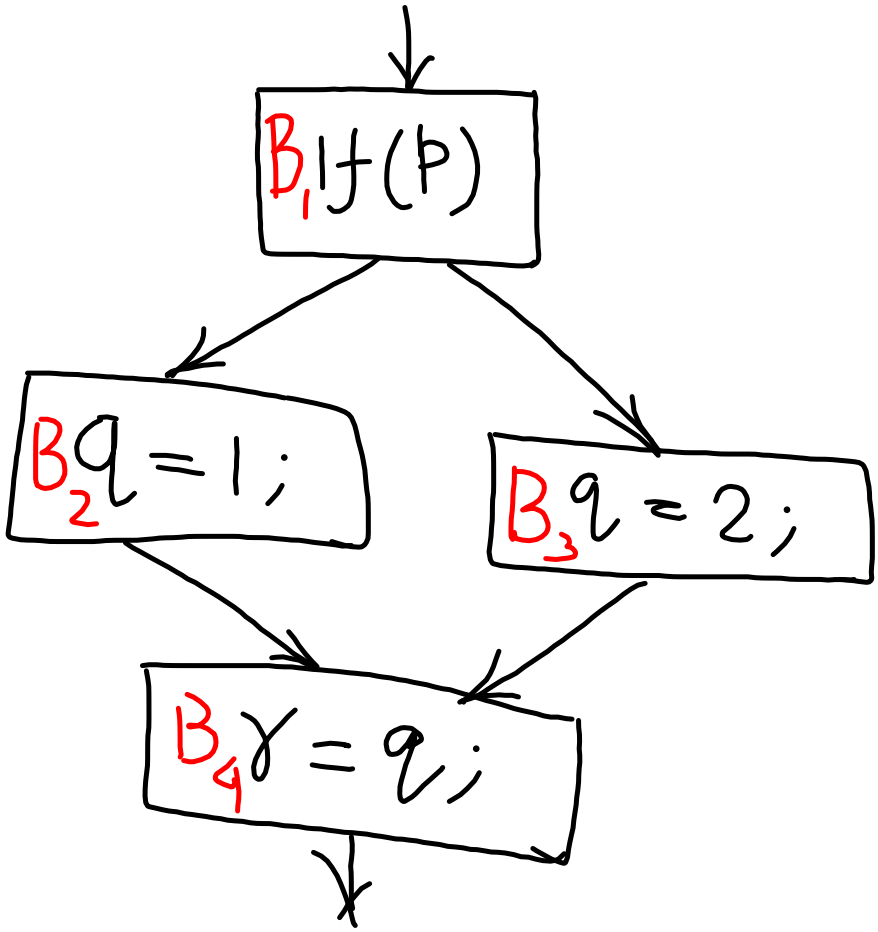
```
while (a < b) {
        a = proc1(a,b);
        b = proc2(a,b);
}
```

C code



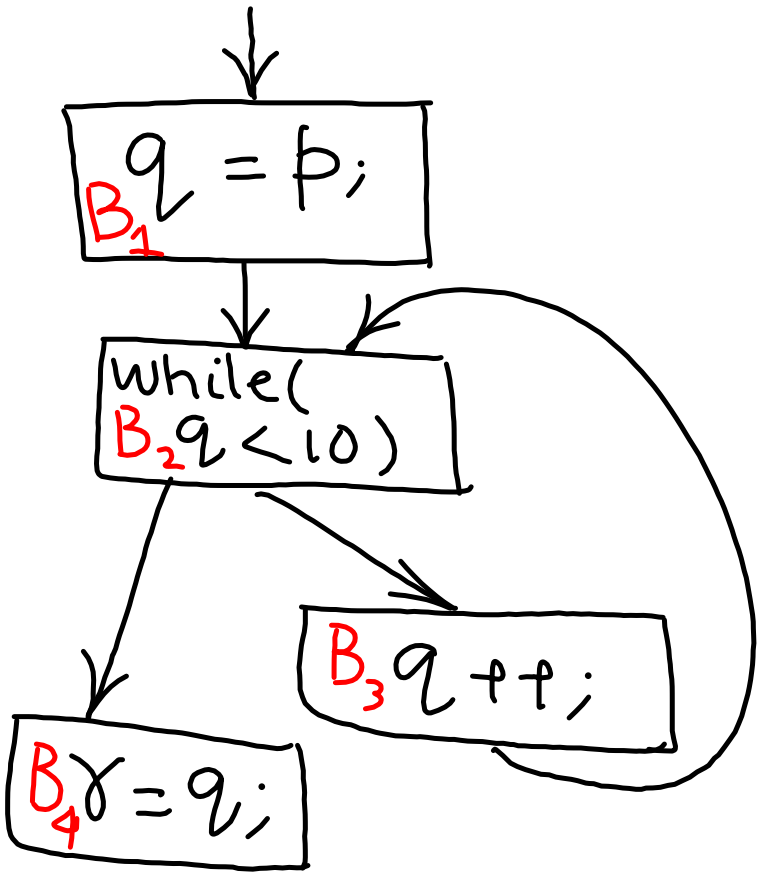
CDFG

```
if (p)
  q = 1;
else
  q = 2;
r = q;
```



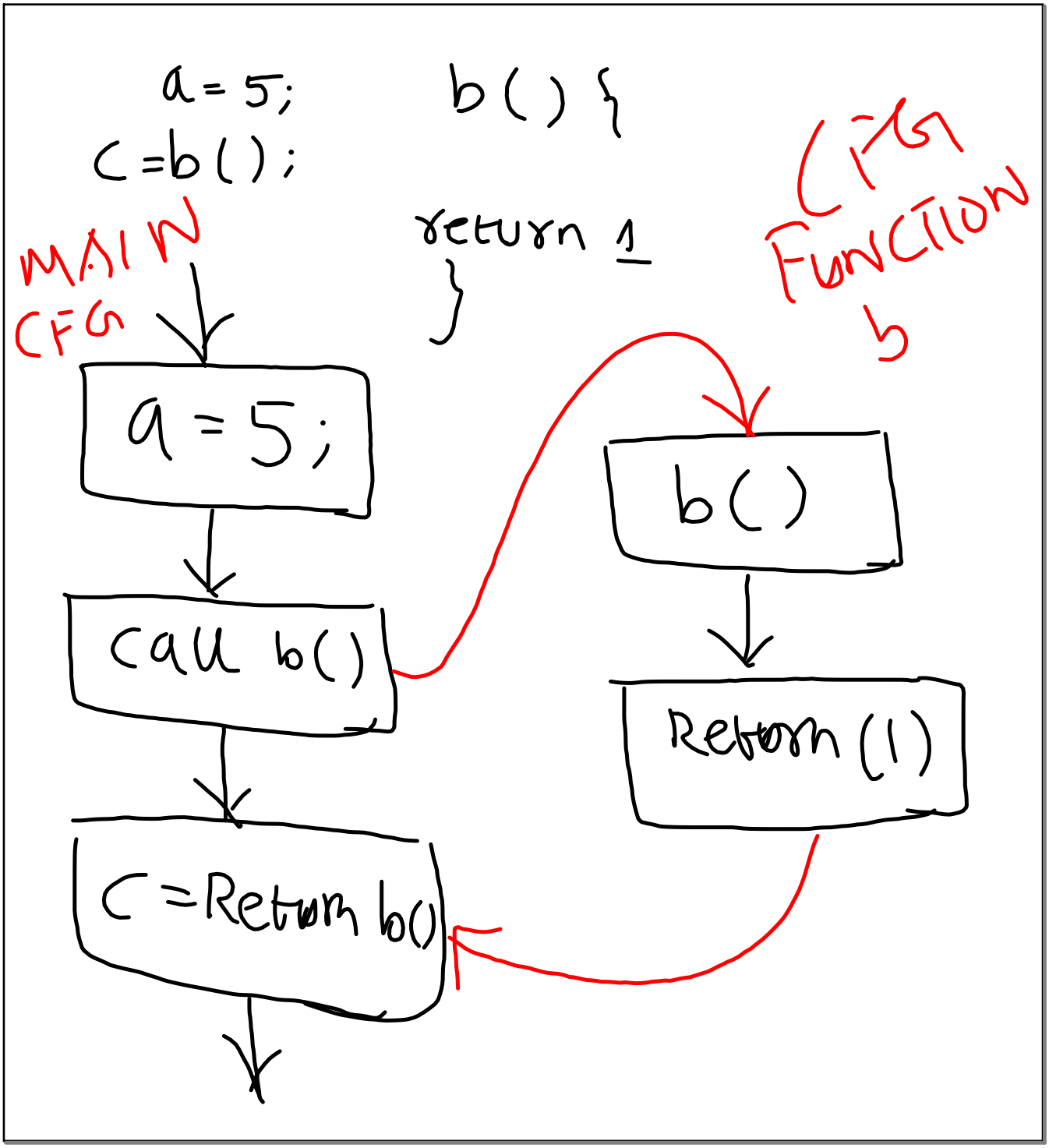
Aug 31-15:51

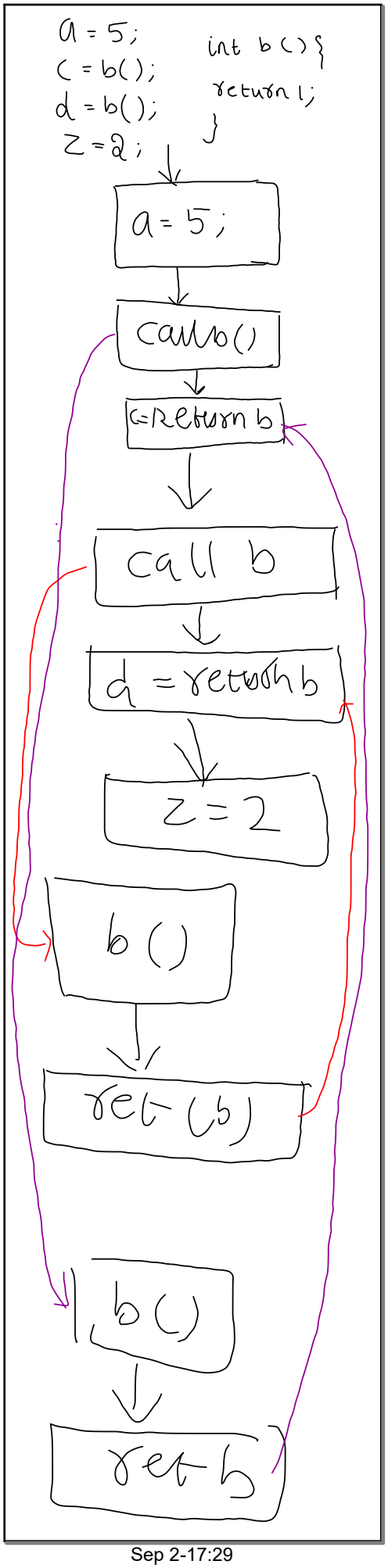
```
/* p >= 0 */
q = p;
while(q<10)
  q++;
r = q;
```



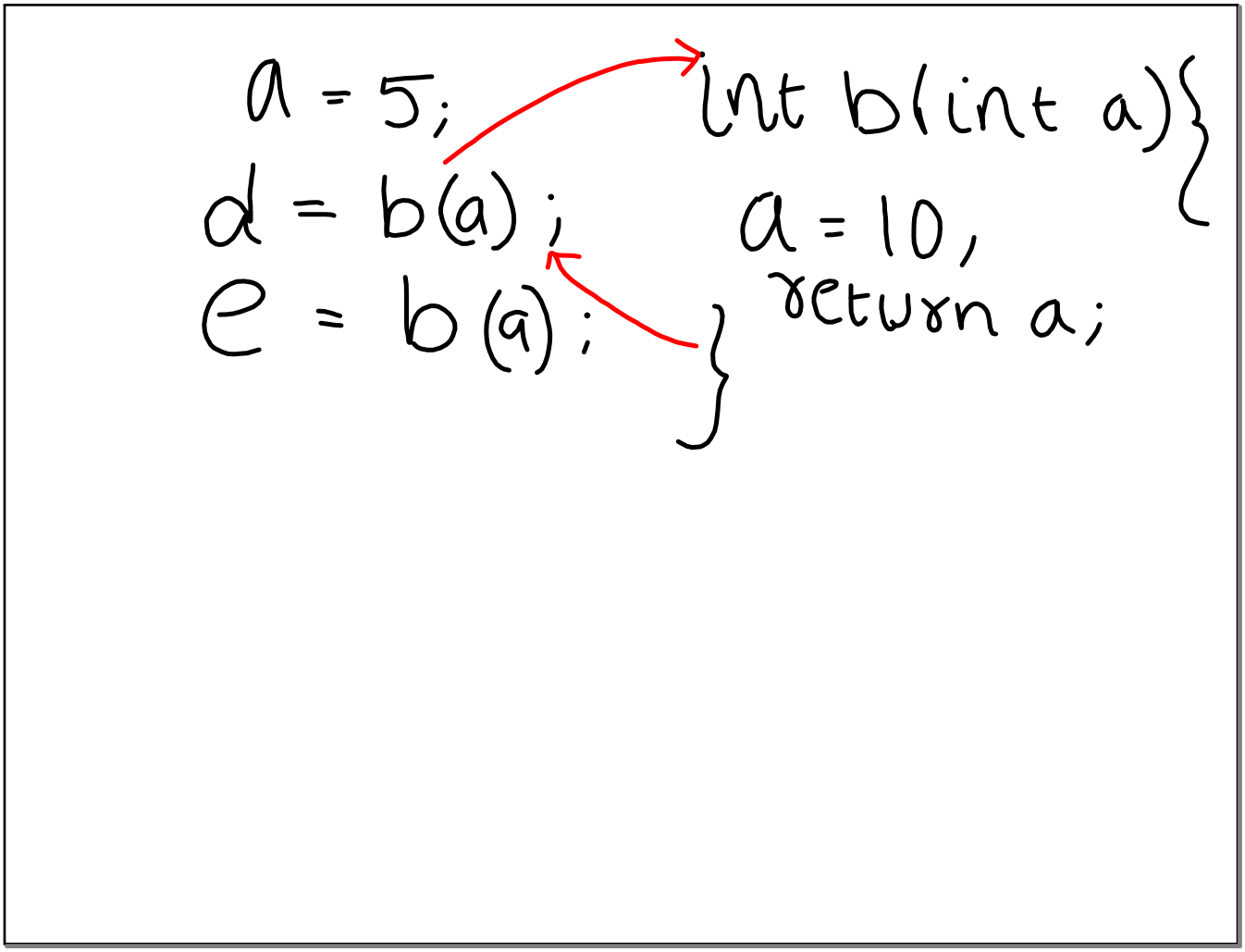
```
i = 10;
store(i);
n = 2*i;
store(n);

void store(int i)
{
   ...
}
```

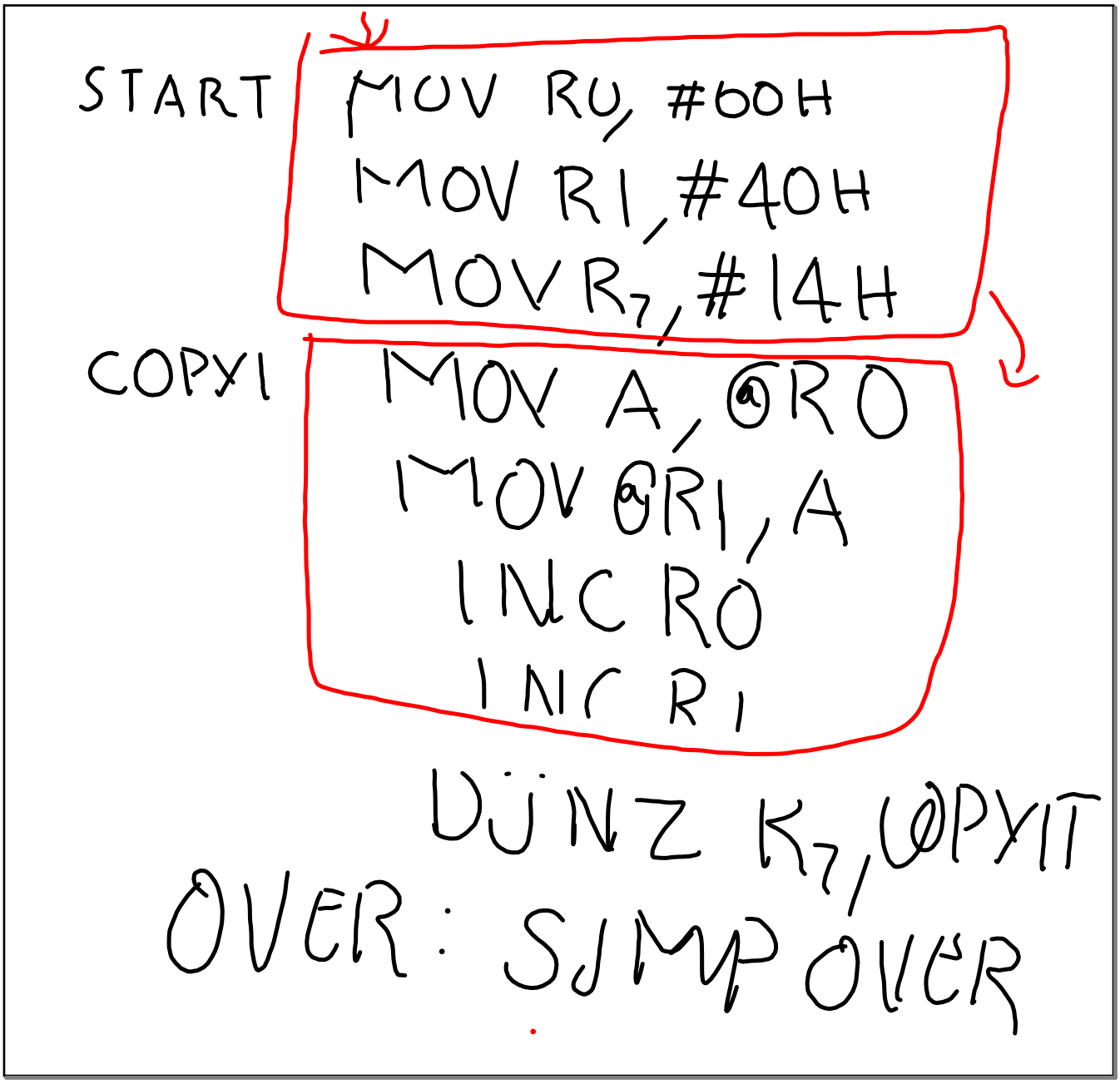
Aug 31-15:52


a = 5;
c = b();
MAIN
CFG
b() {
return 1
}
CFG
FUNCTION
b
a = 5;
call b()
c = Return b()
b()
Return (1)


a = 5;
c = b();
d = b();
z = 2;
int b(){
return 1;
}
a = 5;
call b()
c=return b
call b
d = return b
z = 2
b()
ret (b)
b()
ret b

Sep 2-17:29


a = 5;
d = b(a);
e = b(a);
int b(int a){
a = 10,
return a;
}


START
MOV R0, #60H
MOV R1, #40H
MOV R7, #14H
COPY1
MOV A, @R0
MOV @R1, A
INC R0
INC R1
DJNZ R7, COPY1
OVER: SJMP OVER