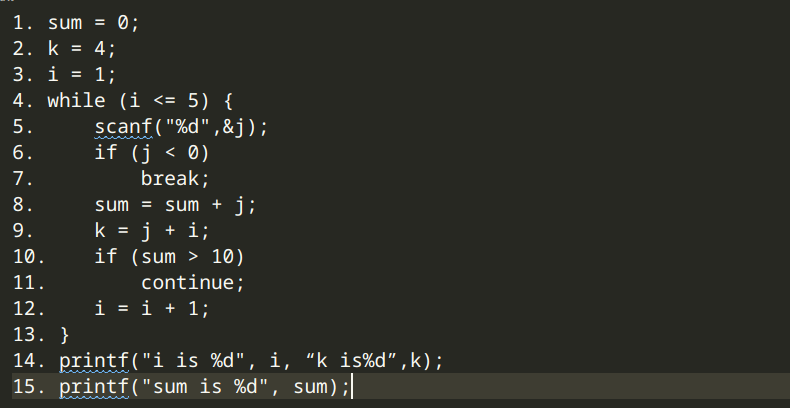
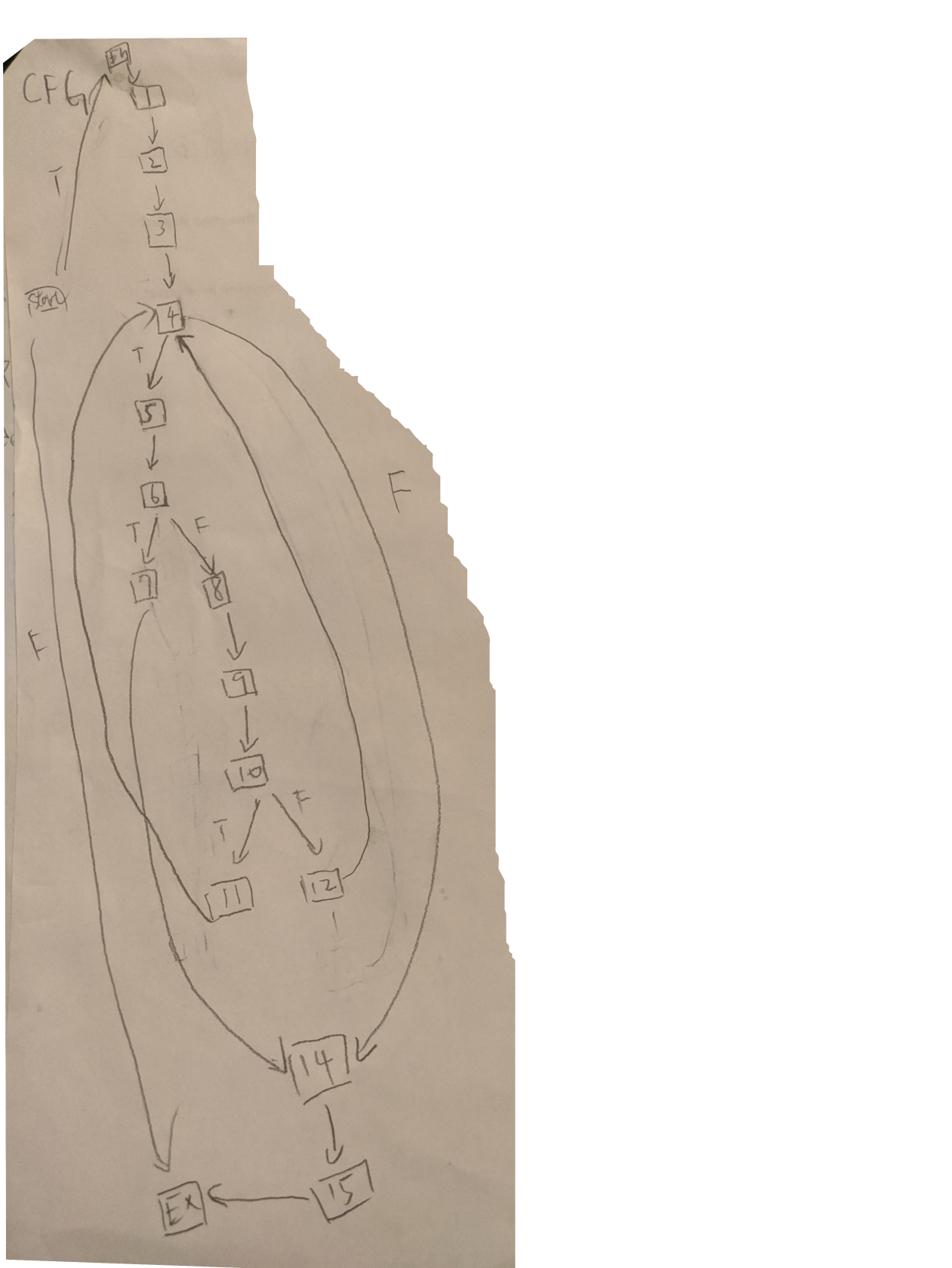
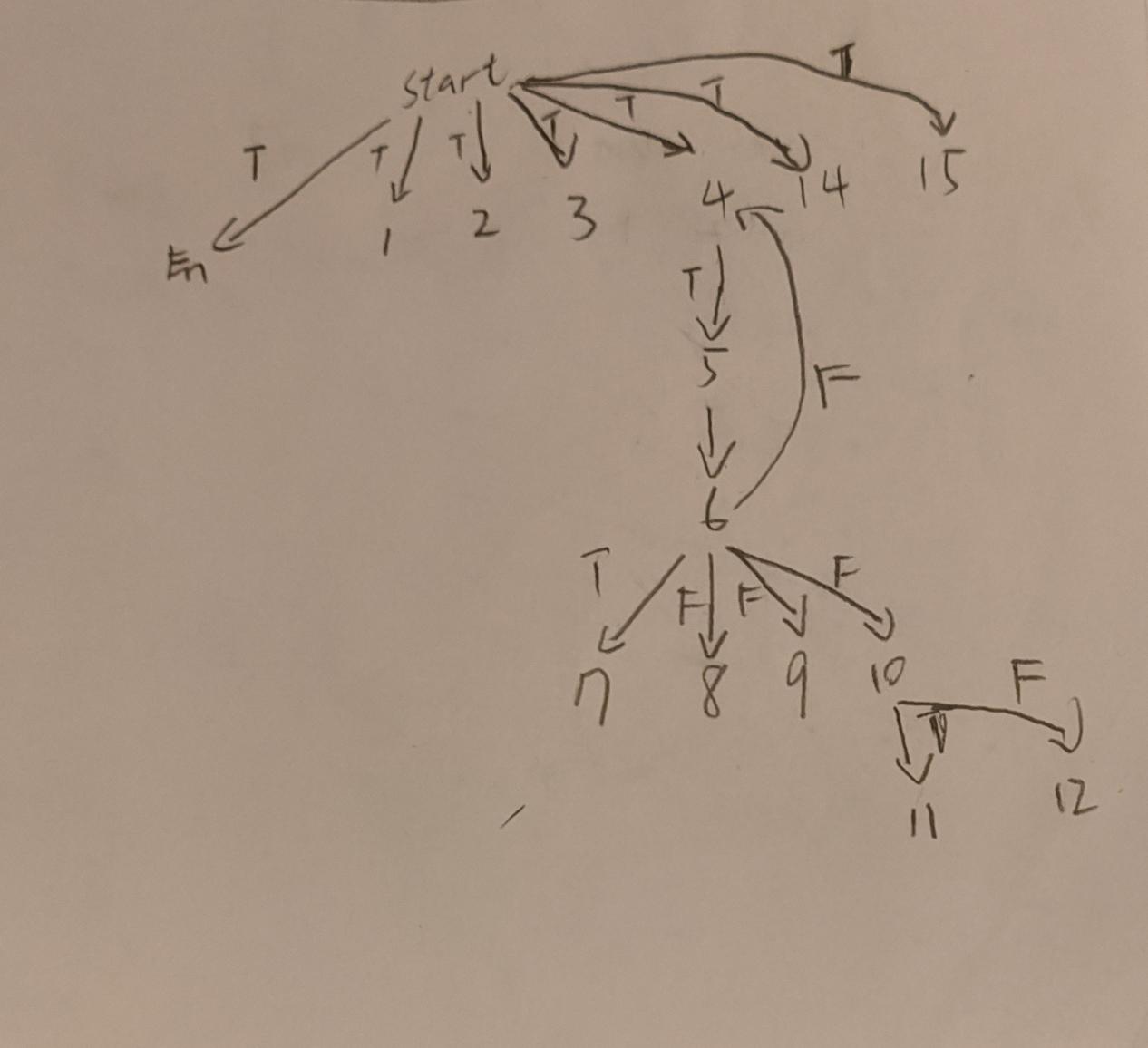
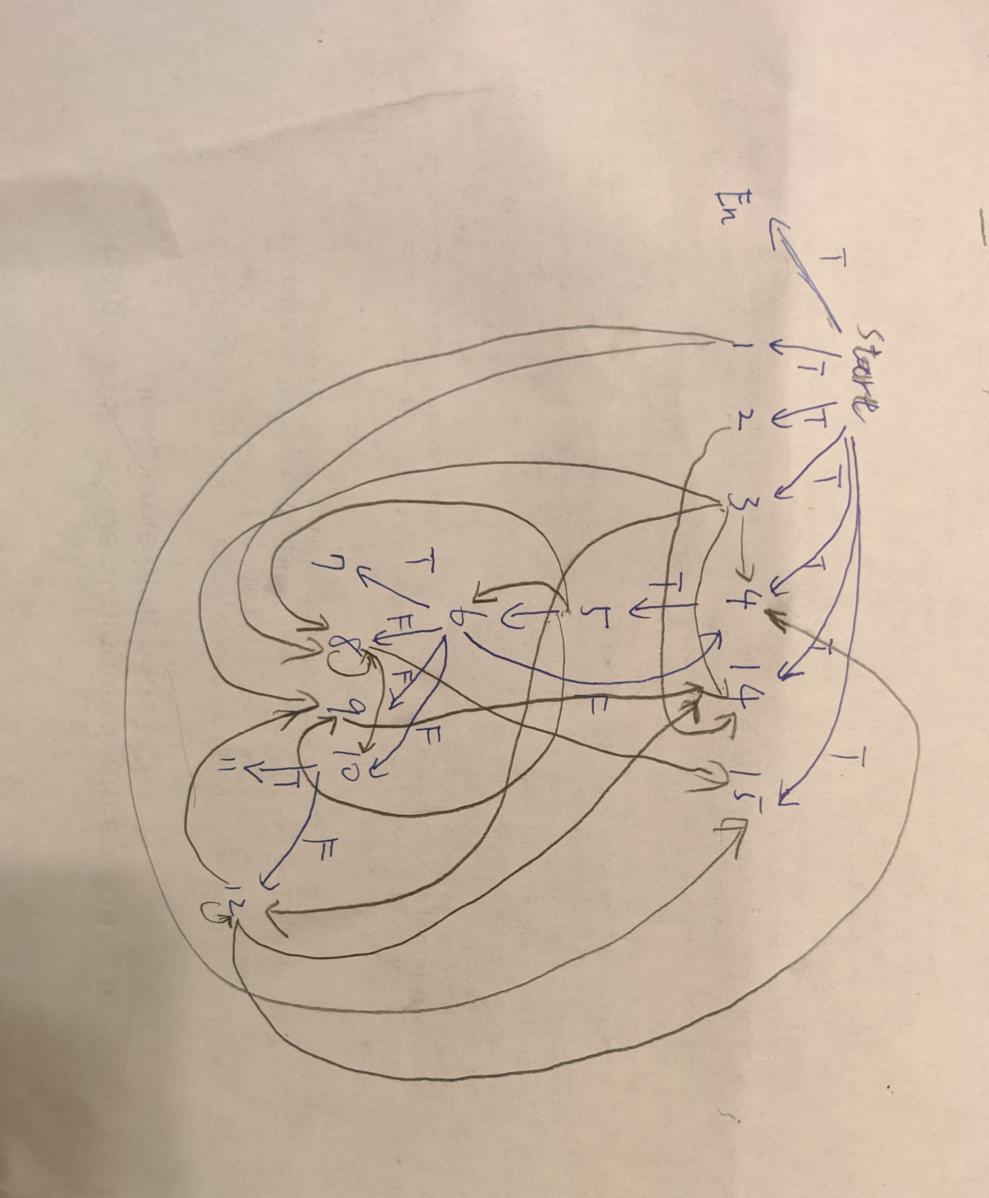
1. compute the statement-based CFG  
   Numbered Code:  
     
   Control Flow Graph  
   
2. construct the control-dependence graph  
   
3. construct the program-dependence graph (PDG)  
   

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n | statement | refs | defs | relevant |
| 1 | sum = 0; |  | sum |  |
| 2 | k = 4; |  | k |  |
| 3 | i = 1; |  | i |  |
| 4 | while (i <= 5) { | i |  | i |
| 5 | scanf("%d",&j); |  | j | i |
| 6 | if (j < 0) | j |  | i |
| 7 | break; |  |  | i |
| 8 | sum = sum + j; | sum, j | sum | i |
| 9 | k = j + i; | i, j | k | i |
| 10 | if (sum > 10) | sum |  | i |
| 11 | continue; |  |  | i |
| 12 | i = i + 1; | i | i | i |
| 14 | printf("i is %d", i, “k is%d”,k); | i, k |  | i |

1. using the CFG, perform slicing on statement 12 for i; show intermediate sets for each iteration.  
     
   Slice on <14, i>: {12, 11, 10, 8, 7, 6, 5, 4, 3, 1}

5. using the PDG, perform slicing on statement 12 for i. Show the steps.

