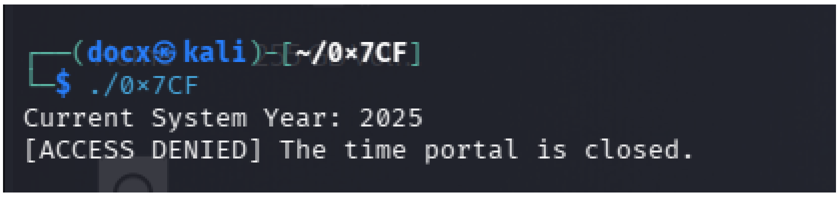
0x7CF

We have a binary on trying to run it normally it says



```
┌──(docx㉿kali)-[~/0x7CF]
└─$ ./0x7CF
Current System Year: 2025
[ACCESS DENIED] The time portal is closed.
```

so we have to either change the system time since it is mentioned in the overview as:

My logic is frozen in a specific frame of reference.
If you query me now, I am silent; my memory is fragmented,
built instruction by instruction on the
stack to hide from your prying eyes.

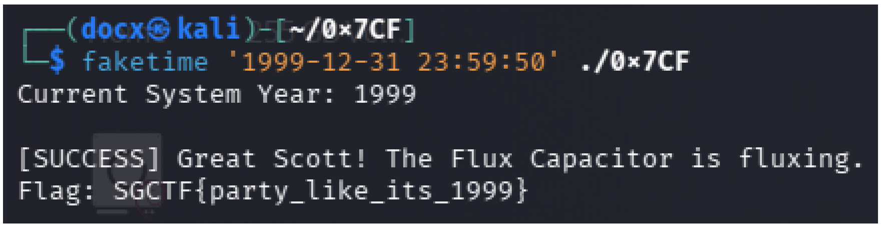
I require a specific environment to function—a temporal
coordinate where the fear of the null byte gripped the world.
Synchronize your watches.

The countdown ends where the zeros begin.

‘The fear of the nullbyte gripped the world’

this line is important as in saying it is talking about the date 1999-12-31
so we can either fake the time or use a debugging tool to find the flag

first we will try the fake time



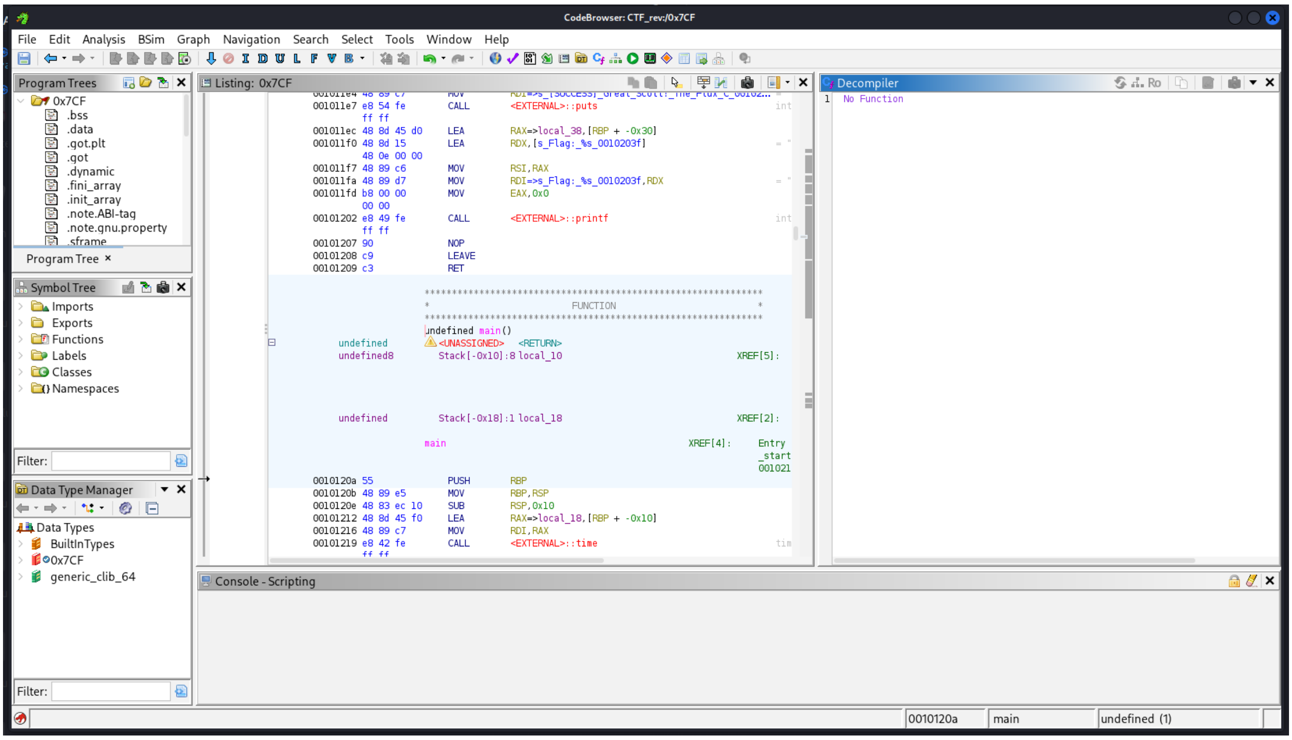
```
┌──(docx㉿kali)-[~/0x7CF]
└─$ faketime '1999-12-31 23:59:50' ./0x7CF
Current System Year: 1999

[SUCCESS] Great Scott! The Flux Capacitor is fluxing.
Flag: SGCTF{party_like_its_1999}
```

which gets us the flag
SGCTF{party_like_its_1999}

now we can try the debugging tool way

import the binary file to ghidra project folder and analyse it with default selections



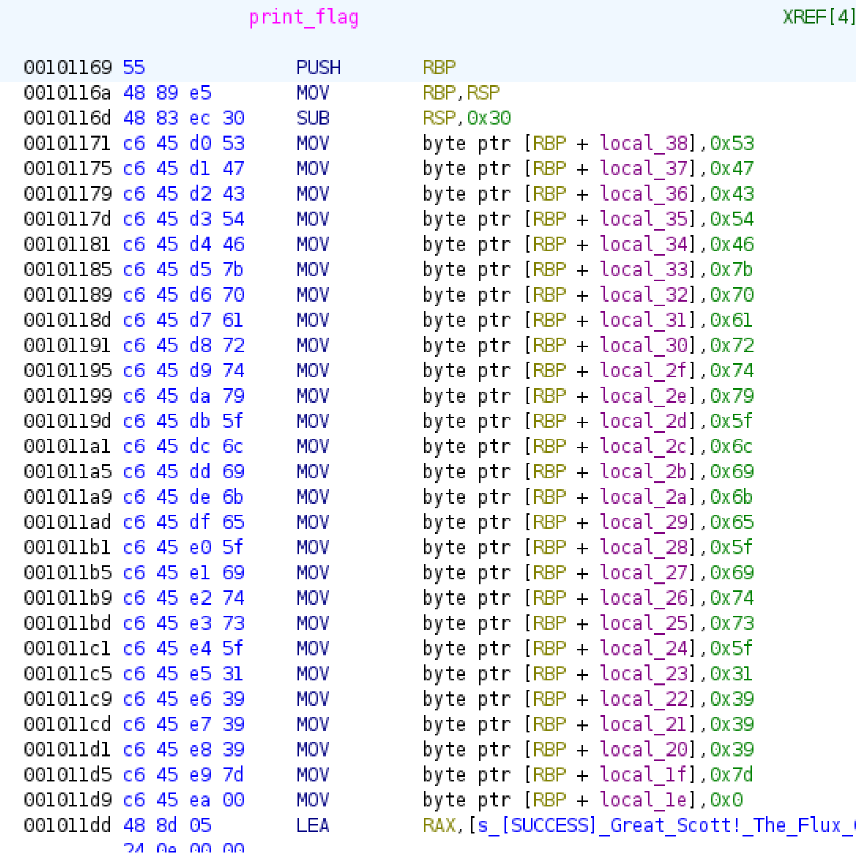
```
        001011e7 e8 54 fe        CALL       <EXTERNAL>::puts                      int
                 ff ff
        001011ec 48 8d 45 d0     LEA        RAX=>local_38,[RBP + -0x30]
        001011f0 48 8d 15        LEA        RDX,[s_Flag:_%s_0010203f]             = "
                 48 0e 00 00
        001011f7 48 89 c6        MOV        RSI,RAX
        001011fa 48 89 d7        MOV        RDI=>s_Flag:_%s_0010203f,RDX          = "
        001011fd b8 00 00        MOV        EAX,0x0
                 00 00
        00101202 e8 49 fe        CALL       <EXTERNAL>::printf                    int
                 ff ff
        00101207 90              NOP
        00101208 c9              LEAVE
        00101209 c3              RET

                          **************************************************************
                          *                          FUNCTION                          *
                          **************************************************************
                          undefined main()
             undefined      <UNASSIGNED>   <RETURN>
             undefined8       Stack[-0x10]:8 local_10                           XREF[5]:


             undefined        Stack[-0x18]:1 local_18                           XREF[2]:

                          main                                          XREF[4]:     Entry
                                                                                     _start
                                                                                     001021
        0010120a 55              PUSH       RBP
        0010120b 48 89 e5        MOV        RBP,RSP
        0010120e 48 83 ec 10     SUB        RSP,0x10
        00101212 48 8d 45 f0     LEA        RAX=>local_18,[RBP + -0x10]
        00101216 48 89 c7        MOV        RDI,RAX
        00101219 e8 42 fe        CALL       <EXTERNAL>::time                      tim
                 ff ff
```

we find the print_flag function which has a series of bytes



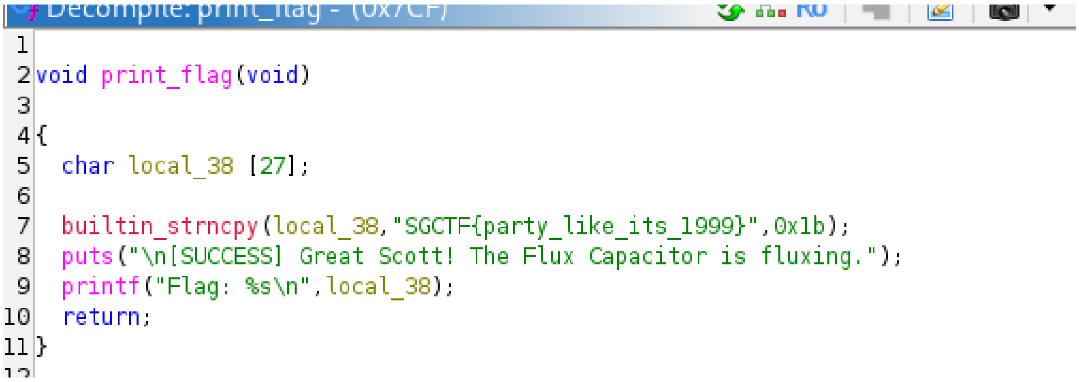
```
                     print_flag                                   XREF[4]

    00101169 55             PUSH     RBP
    0010116a 48 89 e5       MOV      RBP,RSP
    0010116d 48 83 ec 30    SUB      RSP,0x30
    00101171 c6 45 d0 53    MOV      byte ptr [RBP + local_38],0x53
    00101175 c6 45 d1 47    MOV      byte ptr [RBP + local_37],0x47
    00101179 c6 45 d2 43    MOV      byte ptr [RBP + local_36],0x43
    0010117d c6 45 d3 54    MOV      byte ptr [RBP + local_35],0x54
    00101181 c6 45 d4 46    MOV      byte ptr [RBP + local_34],0x46
    00101185 c6 45 d5 7b    MOV      byte ptr [RBP + local_33],0x7b
    00101189 c6 45 d6 70    MOV      byte ptr [RBP + local_32],0x70
    0010118d c6 45 d7 61    MOV      byte ptr [RBP + local_31],0x61
    00101191 c6 45 d8 72    MOV      byte ptr [RBP + local_30],0x72
    00101195 c6 45 d9 74    MOV      byte ptr [RBP + local_2f],0x74
    00101199 c6 45 da 79    MOV      byte ptr [RBP + local_2e],0x79
    0010119d c6 45 db 5f    MOV      byte ptr [RBP + local_2d],0x5f
    001011a1 c6 45 dc 6c    MOV      byte ptr [RBP + local_2c],0x6c
    001011a5 c6 45 dd 69    MOV      byte ptr [RBP + local_2b],0x69
    001011a9 c6 45 de 6b    MOV      byte ptr [RBP + local_2a],0x6b
    001011ad c6 45 df 65    MOV      byte ptr [RBP + local_29],0x65
    001011b1 c6 45 e0 5f    MOV      byte ptr [RBP + local_28],0x5f
    001011b5 c6 45 e1 69    MOV      byte ptr [RBP + local_27],0x69
    001011b9 c6 45 e2 74    MOV      byte ptr [RBP + local_26],0x74
    001011bd c6 45 e3 73    MOV      byte ptr [RBP + local_25],0x73
    001011c1 c6 45 e4 5f    MOV      byte ptr [RBP + local_24],0x5f
    001011c5 c6 45 e5 31    MOV      byte ptr [RBP + local_23],0x31
    001011c9 c6 45 e6 39    MOV      byte ptr [RBP + local_22],0x39
    001011cd c6 45 e7 39    MOV      byte ptr [RBP + local_21],0x39
    001011d1 c6 45 e8 39    MOV      byte ptr [RBP + local_20],0x39
    001011d5 c6 45 e9 7d    MOV      byte ptr [RBP + local_1f],0x7d
    001011d9 c6 45 ea 00    MOV      byte ptr [RBP + local_1e],0x0
    001011dd 48 8d 05       LEA      RAX,[s_[SUCCESS]_Great_Scott!_The_Flux_
             24 0e 00 00
```

select local_38 in it and select retype variable in ghidra

and type char[27]

which turns into



```
void print_flag(void)

{
  char local_38 [27];

  builtin_strncpy(local_38,"SGCTF{party_like_its_1999}",0x1b);
  puts("\n[SUCCESS] Great Scott! The Flux Capacitor is fluxing.");
  printf("Flag: %s\n",local_38);
  return;
}
```

which gives us the flag SGCTF{party_like_its_1999}