```
.plt:000000000040060B                jmp     sub_4005E0
.plt:0000000000400610 ; [00000006 BYTES: COLLAPSED FUNCTION _printf. PRESS CTRL-NUMPAD+ TO EXPAND]
.plt:0000000000400616 ; ---------------------------------------------------------------------------
.plt:0000000000400616                push    2
.plt:000000000040061B                jmp     sub_4005E0
.plt:0000000000400620 ; [00000006 BYTES: COLLAPSED FUNCTION _fgets. PRESS CTRL-NUMPAD+ TO EXPAND]
.plt:0000000000400626 ; ---------------------------------------------------------------------------
.plt:0000000000400626                push    3
.plt:000000000040062B                jmp     sub_4005E0
.plt:0000000000400630 ; [00000006 BYTES: COLLAPSED FUNCTION _strcmp. PRESS CTRL-NUMPAD+ TO EXPAND]
.plt:0000000000400636 ; ---------------------------------------------------------------------------
.plt:0000000000400636                push    4
.plt:000000000040063B                jmp     sub_4005E0
.plt:0000000000400640 ; [00000006 BYTES: COLLAPSED FUNCTION _setvbuf. PRESS CTRL-NUMPAD+ TO EXPAND]
.plt:0000000000400646 ; ---------------------------------------------------------------------------
.plt:0000000000400646                push    5
.plt:000000000040064B                jmp     sub_4005E0
.plt:000000000040064B _plt           ends
.plt:000000000040064B
.text:0000000000400650 ; ===========================================================================
.text:0000000000400650
.text:0000000000400650 ; Segment type: Pure code
.text:0000000000400650 ; Segment permissions: Read/Execute
.text:0000000000400650 _text           segment para public 'CODE' use64
.text:0000000000400650                 assume cs:_text
.text:0000000000400650                 ;org 400650h
.text:0000000000400650                 assume es:nothing, ss:nothing, ds:_data, fs:nothing, gs:nothing
.text:0000000000400650
.text:0000000000400650 ; =============== S U B R O U T I N E =======================================
.text:0000000000400650
.text:0000000000400650 ; Attributes: noreturn fuzzy-sp
.text:0000000000400650
.text:0000000000400650                 public _start
.text:0000000000400650 _start          proc near               ; DATA XREF: LOAD:0000000000400018↑o
.text:0000000000400650                 xor     ebp, ebp
.text:0000000000400652                 mov     r9, rdx         ; rtld_fini
.text:0000000000400655                 pop     rsi             ; argc
.text:0000000000400656                 mov     rdx, rsp        ; ubp_av
.text:0000000000400659                 and     rsp, 0FFFFFFFFFFFFFFF0h
.text:000000000040065D                 push    rax
.text:000000000040065E                 push    rsp             ; stack_end
.text:000000000040065F                 mov     r8, offset __libc_csu_fini ; fini
.text:0000000000400666                 mov     rcx, offset __libc_csu_init ; init
.text:000000000040066D                 mov     rdi, offset main ; main
.text:0000000000400674                 call    cs:__libc_start_main_ptr
.text:000000000040067A                 hlt
.text:000000000040067A _start          endp
.text:000000000040067A
.text:000000000040067A ; ---------------------------------------------------------------------------
.text:000000000040067B                 align 20h
.text:0000000000400680
.text:0000000000400680 ; =============== S U B R O U T I N E =======================================
.text:0000000000400680
.text:0000000000400680 ; Attributes: bp-based frame
.text:0000000000400680
.text:0000000000400680 deregister_tm_clones proc near          ; CODE XREF: __do_global_dtors_aux+D↓p
.text:0000000000400680                 push    rbp
.text:0000000000400681                 mov     eax, offset __TMC_END__
.text:0000000000400686                 cmp     rax, offset __TMC_END__
.text:000000000040068C                 mov     rbp, rsp
.text:000000000040068F                 jz      short loc_4006A8
.text:0000000000400691                 mov     eax, 0
.text:0000000000400696                 test    rax, rax
.text:0000000000400699                 jz      short loc_4006A8
.text:000000000040069B                 pop     rbp
.text:000000000040069C                 mov     edi, offset __TMC_END__
.text:00000000004006A1                 jmp     rax
.text:00000000004006A1 ; ---------------------------------------------------------------------------
.text:00000000004006A3                 align 8
.text:00000000004006A8
.text:00000000004006A8 loc_4006A8:                             ; CODE XREF: deregister_tm_clones+F↑j
.text:00000000004006A8                                         ; deregister_tm_clones+19↑j
.text:00000000004006A8                 pop     rbp
.text:00000000004006A9                 retn
.text:00000000004006A9 deregister_tm_clones endp
.text:00000000004006A9
.text:00000000004006A9 ; ---------------------------------------------------------------------------
.text:00000000004006AA                 align 10h
.text:00000000004006B0
.text:00000000004006B0 ; =============== S U B R O U T I N E =======================================
.text:00000000004006B0
.text:00000000004006B0 ; Attributes: bp-based frame
.text:00000000004006B0
.text:00000000004006B0 register_tm_clones proc near            ; CODE XREF: frame_dummy+5↓j
.text:00000000004006B0                 mov     esi, offset __TMC_END__
.text:00000000004006B5                 push    rbp
.text:00000000004006B6                 sub     rsi, offset __TMC_END__
.text:00000000004006BD                 mov     rbp, rsp
.text:00000000004006C0                 sar     rsi, 3
.text:00000000004006C4                 mov     rax, rsi
```

```
.text:00000000004006C7                 shr     rax, 3Fh
.text:00000000004006CB                 add     rsi, rax
.text:00000000004006CE                 sar     rsi, 1
.text:00000000004006D1                 jz      short loc_4006E8
.text:00000000004006D3                 mov     eax, 0
.text:00000000004006D8                 test    rax, rax
.text:00000000004006DB                 jz      short loc_4006E8
.text:00000000004006DD                 pop     rbp
.text:00000000004006DE                 mov     edi, offset __TMC_END__
.text:00000000004006E3                 jmp     rax
.text:00000000004006E3 ; ---------------------------------------------------------------------------
.text:00000000004006E5                 align 8
.text:00000000004006E8
.text:00000000004006E8 loc_4006E8:                             ; CODE XREF: register_tm_clones+21↑j
.text:00000000004006E8                                         ; register_tm_clones+2B↑j
.text:00000000004006E8                 pop     rbp
.text:00000000004006E9                 retn
.text:00000000004006E9 register_tm_clones endp
.text:00000000004006E9
.text:00000000004006E9 ; ---------------------------------------------------------------------------
.text:00000000004006EA                 align 10h
.text:00000000004006F0
.text:00000000004006F0 ; =============== S U B R O U T I N E =======================================
.text:00000000004006F0
.text:00000000004006F0
.text:00000000004006F0 __do_global_dtors_aux proc near         ; DATA XREF: .fini_array:__do_global_dtors_aux_fini_array_entry↓o
.text:00000000004006F0                 cmp     cs:completed_6984, 0
.text:00000000004006F7                 jnz     short locret_400710
.text:00000000004006F9                 push    rbp
.text:00000000004006FA                 mov     rbp, rsp
.text:00000000004006FD                 call    deregister_tm_clones
.text:0000000000400702                 mov     cs:completed_6984, 1
.text:0000000000400709                 pop     rbp
.text:000000000040070A                 retn
.text:000000000040070A ; ---------------------------------------------------------------------------
.text:000000000040070B                 align 10h
.text:0000000000400710
.text:0000000000400710 locret_400710:                          ; CODE XREF: __do_global_dtors_aux+7↑j
.text:0000000000400710                 rep retn
.text:0000000000400710 __do_global_dtors_aux endp
.text:0000000000400710
.text:0000000000400710 ; ---------------------------------------------------------------------------
.text:0000000000400712                 align 20h
.text:0000000000400720
.text:0000000000400720 ; =============== S U B R O U T I N E =======================================
.text:0000000000400720
.text:0000000000400720 ; Attributes: bp-based frame
.text:0000000000400720
.text:0000000000400720 frame_dummy     proc near               ; DATA XREF: .init_array:__frame_dummy_init_array_entry↓o
.text:0000000000400720                 push    rbp
.text:0000000000400721                 mov     rbp, rsp
.text:0000000000400724                 pop     rbp
.text:0000000000400725                 jmp     short register_tm_clones
.text:0000000000400725 frame_dummy     endp
.text:0000000000400725
.text:0000000000400727
.text:0000000000400727 ; =============== S U B R O U T I N E =======================================
.text:0000000000400727
.text:0000000000400727 ; Attributes: bp-based frame
.text:0000000000400727
.text:0000000000400727                 public echo
.text:0000000000400727 echo            proc near               ; CODE XREF: main+6E↓p
.text:0000000000400727
.text:0000000000400727 var_118         = qword ptr -118h
.text:0000000000400727 s               = byte ptr -110h
.text:0000000000400727 var_8           = qword ptr -8
.text:0000000000400727
.text:0000000000400727                 push    rbp
.text:0000000000400728                 mov     rbp, rsp
.text:000000000040072B                 sub     rsp, 120h
.text:0000000000400732                 mov     rax, fs:28h
.text:000000000040073B                 mov     [rbp+var_8], rax
.text:000000000040073F                 xor     eax, eax
.text:0000000000400741                 mov     edi, offset s   ; "200 OK ECHO (v0.2)"
.text:0000000000400746                 call    _puts
.text:000000000040074B
.text:000000000040074B loc_40074B:                             ; CODE XREF: echo+7C↓j
.text:000000000040074B                 mov     rdx, cs:stdin@@GLIBC_2_2_5 ; stream
.text:0000000000400752                 lea     rax, [rbp+s]
.text:0000000000400759                 mov     esi, 100h       ; n
.text:000000000040075E                 mov     rdi, rax        ; s
.text:0000000000400761                 call    _fgets
.text:0000000000400766                 mov     [rbp+var_118], rax
.text:000000000040076D                 cmp     [rbp+var_118], 0
.text:0000000000400775                 jz      short loc_4007A5
.text:0000000000400777                 lea     rax, [rbp+s]
.text:000000000040077E                 mov     rdi, rax        ; format
.text:0000000000400781                 mov     eax, 0
.text:0000000000400786                 call    _printf
.text:000000000040078B                 lea     rax, [rbp+s]
```

```
.text:0000000000400792                 mov     esi, offset s2  ; "quit\n"
.text:0000000000400797                 mov     rdi, rax        ; s1
.text:000000000040079A                 call    _strcmp
.text:000000000040079F                 test    eax, eax
.text:00000000004007A1                 jz      short loc_4007A8
.text:00000000004007A3                 jmp     short loc_40074B
.text:00000000004007A5 ; ---------------------------------------------------------------------------
.text:00000000004007A5
.text:00000000004007A5 loc_4007A5:                             ; CODE XREF: echo+4E↑j
.text:00000000004007A5                 nop
.text:00000000004007A6                 jmp     short loc_4007A9
.text:00000000004007A8 ; ---------------------------------------------------------------------------
.text:00000000004007A8
.text:00000000004007A8 loc_4007A8:                             ; CODE XREF: echo+7A↑j
.text:00000000004007A8                 nop
.text:00000000004007A9
.text:00000000004007A9 loc_4007A9:                             ; CODE XREF: echo+7F↑j
.text:00000000004007A9                 nop
.text:00000000004007AA                 mov     rax, [rbp+var_8]
.text:00000000004007AE                 xor     rax, fs:28h
.text:00000000004007B7                 jz      short locret_4007BE
.text:00000000004007B9                 call    ___stack_chk_fail
.text:00000000004007BE
.text:00000000004007BE locret_4007BE:                          ; CODE XREF: echo+90↑j
.text:00000000004007BE                 leave
.text:00000000004007BF                 retn
.text:00000000004007BF echo            endp
.text:00000000004007BF
.text:00000000004007C0
.text:00000000004007C0 ; =============== S U B R O U T I N E =======================================
.text:00000000004007C0
.text:00000000004007C0 ; Attributes: bp-based frame
.text:00000000004007C0
.text:00000000004007C0 ; int __cdecl main(int argc, const char **argv, const char **envp)
.text:00000000004007C0                 public main
.text:00000000004007C0 main            proc near               ; DATA XREF: _start+1D↑o
.text:00000000004007C0
.text:00000000004007C0 var_10          = qword ptr -10h
.text:00000000004007C0 var_4           = dword ptr -4
.text:00000000004007C0
.text:00000000004007C0                 push    rbp
.text:00000000004007C1                 mov     rbp, rsp
.text:00000000004007C4                 sub     rsp, 10h
.text:00000000004007C8                 mov     [rbp+var_4], edi
.text:00000000004007CB                 mov     [rbp+var_10], rsi
.text:00000000004007CF                 mov     rax, cs:stdin@@GLIBC_2_2_5
.text:00000000004007D6                 mov     ecx, 0          ; n
.text:00000000004007DB                 mov     edx, 2          ; modes
.text:00000000004007E0                 mov     esi, 0          ; buf
.text:00000000004007E5                 mov     rdi, rax        ; stream
.text:00000000004007E8                 call    _setvbuf
.text:00000000004007ED                 mov     rax, cs:stdout@@GLIBC_2_2_5
.text:00000000004007F4                 mov     ecx, 0          ; n
.text:00000000004007F9                 mov     edx, 2          ; modes
.text:00000000004007FE                 mov     esi, 0          ; buf
.text:0000000000400803                 mov     rdi, rax        ; stream
.text:0000000000400806                 call    _setvbuf
.text:000000000040080B                 mov     rax, cs:stderr@@GLIBC_2_2_5
.text:0000000000400812                 mov     ecx, 0          ; n
.text:0000000000400817                 mov     edx, 2          ; modes
.text:000000000040081C                 mov     esi, 0          ; buf
.text:0000000000400821                 mov     rdi, rax        ; stream
.text:0000000000400824                 call    _setvbuf
.text:0000000000400829                 mov     eax, 0
.text:000000000040082E                 call    echo
.text:0000000000400833                 mov     eax, 0
.text:0000000000400838                 leave
.text:0000000000400839                 retn
.text:0000000000400839 main            endp
.text:0000000000400839
.text:0000000000400839 ; ---------------------------------------------------------------------------
.text:000000000040083A                 align 20h
.text:0000000000400840
.text:0000000000400840 ; =============== S U B R O U T I N E =======================================
.text:0000000000400840
.text:0000000000400840
.text:0000000000400840 ; void _libc_csu_init(void)
.text:0000000000400840                 public __libc_csu_init
.text:0000000000400840 __libc_csu_init proc near               ; DATA XREF: _start+16↑o
.text:0000000000400840                 push    r15
.text:0000000000400842                 push    r14
.text:0000000000400844                 mov     r15, rdx
.text:0000000000400847                 push    r13
.text:0000000000400849                 push    r12
.text:000000000040084B                 lea     r12, __frame_dummy_init_array_entry
.text:0000000000400852                 push    rbp
.text:0000000000400853                 lea     rbp, __do_global_dtors_aux_fini_array_entry
.text:000000000040085A                 push    rbx
.text:000000000040085B                 mov     r13d, edi
.text:000000000040085E                 mov     r14, rsi
```

```
.text:0000000000400861                 sub     rbp, r12
.text:0000000000400864                 sub     rsp, 8
.text:0000000000400868                 sar     rbp, 3
.text:000000000040086C                 call    _init_proc
.text:0000000000400871                 test    rbp, rbp
.text:0000000000400874                 jz      short loc_400896
.text:0000000000400876                 xor     ebx, ebx
.text:0000000000400878                 nop     dword ptr [rax+rax+00000000h]
.text:0000000000400880
.text:0000000000400880 loc_400880:                             ; CODE XREF: __libc_csu_init+54↓j
.text:0000000000400880                 mov     rdx, r15
.text:0000000000400883                 mov     rsi, r14
.text:0000000000400886                 mov     edi, r13d
.text:0000000000400889                 call    qword ptr [r12+rbx*8]
.text:000000000040088D                 add     rbx, 1
.text:0000000000400891                 cmp     rbp, rbx
.text:0000000000400894                 jnz     short loc_400880
.text:0000000000400896
.text:0000000000400896 loc_400896:                             ; CODE XREF: __libc_csu_init+34↑j
.text:0000000000400896                 add     rsp, 8
.text:000000000040089A                 pop     rbx
.text:000000000040089B                 pop     rbp
.text:000000000040089C                 pop     r12
.text:000000000040089E                 pop     r13
.text:00000000004008A0                 pop     r14
.text:00000000004008A2                 pop     r15
.text:00000000004008A4                 retn
.text:00000000004008A4 __libc_csu_init endp
.text:00000000004008A4
.text:00000000004008A4 ; ---------------------------------------------------------------------------
.text:00000000004008A5                 align 10h
.text:00000000004008B0
.text:00000000004008B0 ; =============== S U B R O U T I N E =======================================
.text:00000000004008B0
.text:00000000004008B0
.text:00000000004008B0 ; void _libc_csu_fini(void)
.text:00000000004008B0                 public __libc_csu_fini
.text:00000000004008B0 __libc_csu_fini proc near               ; DATA XREF: _start+F↑o
.text:00000000004008B0                 rep retn
.text:00000000004008B0 __libc_csu_fini endp
.text:00000000004008B0
.text:00000000004008B0 _text           ends
.text:00000000004008B0
LOAD:00000000004008B2 ; ===========================================================================
LOAD:00000000004008B2
LOAD:00000000004008B2 ; Segment type: Pure code
LOAD:00000000004008B2 ; Segment permissions: Read/Execute
LOAD:00000000004008B2 LOAD            segment byte public 'CODE' use64
LOAD:00000000004008B2                 assume cs:LOAD
LOAD:00000000004008B2                 ;org 4008B2h
LOAD:00000000004008B2                 assume es:nothing, ss:nothing, ds:_data, fs:nothing, gs:nothing
LOAD:00000000004008B2                 align 4
LOAD:00000000004008B2 LOAD            ends
LOAD:00000000004008B2
.fini:00000000004008B4 ; ===========================================================================
.fini:00000000004008B4
.fini:00000000004008B4 ; Segment type: Pure code
.fini:00000000004008B4 ; Segment permissions: Read/Execute
.fini:00000000004008B4 _fini           segment dword public 'CODE' use64
.fini:00000000004008B4                 assume cs:_fini
.fini:00000000004008B4                 ;org 4008B4h
.fini:00000000004008B4                 assume es:nothing, ss:nothing, ds:_data, fs:nothing, gs:nothing
.fini:00000000004008B4
.fini:00000000004008B4 ; =============== S U B R O U T I N E =======================================
.fini:00000000004008B4
.fini:00000000004008B4
.fini:00000000004008B4                 public _term_proc
.fini:00000000004008B4 _term_proc      proc near
.fini:00000000004008B4                 sub     rsp, 8          ; _fini
.fini:00000000004008B8                 add     rsp, 8
.fini:00000000004008BC                 retn
.fini:00000000004008BC _term_proc      endp
.fini:00000000004008BC
.fini:00000000004008BC _fini           ends
.fini:00000000004008BC
LOAD:00000000004008BD ; ===========================================================================
LOAD:00000000004008BD
LOAD:00000000004008BD ; Segment type: Pure code
LOAD:00000000004008BD ; Segment permissions: Read/Execute
LOAD:00000000004008BD LOAD            segment byte public 'CODE' use64
LOAD:00000000004008BD                 assume cs:LOAD
LOAD:00000000004008BD                 ;org 4008BDh
LOAD:00000000004008BD                 assume es:nothing, ss:nothing, ds:_data, fs:nothing, gs:nothing
LOAD:00000000004008BD                 align 20h
LOAD:00000000004008BD LOAD            ends
LOAD:00000000004008BD
.rodata:00000000004008C0 ; ===========================================================================
.rodata:00000000004008C0
.rodata:00000000004008C0 ; Segment type: Pure data
.rodata:00000000004008C0 ; Segment permissions: Read
```