

1. Line 5 (**char globBuf[65536];**) - stored in BSS or uninitialized data because the array of chars is not initialized only declared in the global area of the program.

- Before photo:

```
jessica@jessica-VirtualBox:~/fwork_313628364/q_1$ size
text      data      bss      dec      hex filename
1829      628      1030556      1030805      9d49b9 a.out
jessica@jessica-VirtualBox:~/fwork_313628364/q_1$
```

```
jessica@jessica-VirtualBox:~/fwork_313628364/q_1$ size
text    data    bss      dec      hex filename
1829     628 10240032 10242489 9c49b9 a.out
jessica@jessica-VirtualBox:~/fwork_313628364/q_1$
```

- I used the command `size` to see where the array is allocated. The First time i ran `size` while the array was part of the program and the second time when the array was not part of the program. When it wasn't part of the program I could see that the data wwent down meaning less saved in data and thats how I know it was allocated in data.

The screenshot displays the Visual Studio Code interface with a C program open in the editor and its symbol table visible in the Output window.

**Editor Content:**

```

q1_313628364.c
1  #include <stdio.h>
2  #include <math.h>
3
4  int main()
5  {
6      static int square(int x) /* 3. TEXT */
7      {
8          int result; /* 4. STACK */
9
10         result = x * x;
11     }
12
13     printf("Enter a number: ");
14     int n;
15     scanf("%d", &n);
16     printf("Square of %d is: %d\n", n, square(n));
17     return 0;
18 }

```

**Output Window (TERMINAL):**

```

4: bash

```

**Output Window (PROBLEMS):**

```

Name      Value      Class      Type      Size      Line      Section
-----
_bss_start 0000000000201024 B          NOTYPE    0          |_bss
_completed 0000000000201040 B          NOTYPE    0          |_bss
__cxa_finalize@GLIBC_2.2.5 0000000000201060 d          FUNC      0000000000000001 |__UND*
__data_start 0000000000201080 d          NOTYPE    0          |_data
__data_end 00000000002010A0 d          NOTYPE    0          |_data
__deregister_tm_clones 00000000002010C0 t          FUNC      000000000000005F |_text
__dso_handle 00000000002010E0 t          FUNC      000000000000005F |_text
__do_global_ctors_aux@FINI 0000000000201100 t          FUNC      000000000000005F |_text
__do_global_ctors_aux@FINI 0000000000201120 t          FUNC      000000000000005F |_text
__dynamic 0000000000201140 d          OBJECT    0000000000000000 |_fini_array
__edata 0000000000201160 d          NOTYPE    0          |_data
__eh_frame 0000000000201180 d          OBJECT    0000000000000000 |_eh_frame
__eh_frame_hdr 00000000002011A0 d          OBJECT    0000000000000000 |_eh_frame_hdr
__init_array_end 00000000002011C0 t          FUNC      0000000000000000 |_init_array
__init_array_start 00000000002011E0 t          FUNC      0000000000000000 |_init_array
__rodata_end 0000000000201200 r          NOTYPE    0          |_rodata
__rodata_start 0000000000201220 r          NOTYPE    0          |_rodata
__stack_end 0000000000201240 t          FUNC      0000000000000000 |_stack
__stack_start 0000000000201260 t          FUNC      0000000000000000 |_stack
__tdata_end 0000000000201280 t          FUNC      0000000000000000 |_tdata
__tdata_start 00000000002012A0 t          FUNC      0000000000000000 |_tdata
__unwind_start 00000000002012C0 t          FUNC      0000000000000000 |_unwind
__vtable_start 00000000002012E0 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201300 t          FUNC      0000000000000000 |_vtable
__vtable_start 0000000000201320 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201340 t          FUNC      0000000000000000 |_vtable
__vtable_start 0000000000201360 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201380 t          FUNC      0000000000000000 |_vtable
__vtable_start 00000000002013A0 t          FUNC      0000000000000000 |_vtable
__vtable_end 00000000002013C0 t          FUNC      0000000000000000 |_vtable
__vtable_start 00000000002013E0 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201400 t          FUNC      0000000000000000 |_vtable
__vtable_start 0000000000201420 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201440 t          FUNC      0000000000000000 |_vtable
__vtable_start 0000000000201460 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201480 t          FUNC      0000000000000000 |_vtable
__vtable_start 00000000002014A0 t          FUNC      0000000000000000 |_vtable
__vtable_end 00000000002014C0 t          FUNC      0000000000000000 |_vtable
__vtable_start 00000000002014E0 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201500 t          FUNC      0000000000000000 |_vtable
__vtable_start 0000000000201520 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201540 t          FUNC      0000000000000000 |_vtable
__vtable_start 0000000000201560 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201580 t          FUNC      0000000000000000 |_vtable
__vtable_start 00000000002015A0 t          FUNC      0000000000000000 |_vtable
__vtable_end 00000000002015C0 t          FUNC      0000000000000000 |_vtable
__vtable_start 00000000002015E0 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201600 t          FUNC      0000000000000000 |_vtable
__vtable_start 0000000000201620 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201640 t          FUNC      0000000000000000 |_vtable
__vtable_start 0000000000201660 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201680 t          FUNC      0000000000000000 |_vtable
__vtable_start 00000000002016A0 t          FUNC      0000000000000000 |_vtable
__vtable_end 00000000002016C0 t          FUNC      0000000000000000 |_vtable
__vtable_start 00000000002016E0 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201700 t          FUNC      0000000000000000 |_vtable
__vtable_start 0000000000201720 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201740 t          FUNC      0000000000000000 |_vtable
__vtable_start 0000000000201760 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201780 t          FUNC      0000000000000000 |_vtable
__vtable_start 00000000002017A0 t          FUNC      0000000000000000 |_vtable
__vtable_end 00000000002017C0 t          FUNC      0000000000000000 |_vtable
__vtable_start 00000000002017E0 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201800 t          FUNC      0000000000000000 |_vtable
__vtable_start 0000000000201820 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201840 t          FUNC      0000000000000000 |_vtable
__vtable_start 0000000000201860 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201880 t          FUNC      0000000000000000 |_vtable
__vtable_start 00000000002018A0 t          FUNC      0000000000000000 |_vtable
__vtable_end 00000000002018C0 t          FUNC      0000000000000000 |_vtable
__vtable_start 00000000002018E0 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201900 t          FUNC      0000000000000000 |_vtable
__vtable_start 0000000000201920 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201940 t          FUNC      0000000000000000 |_vtable
__vtable_start 0000000000201960 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201980 t          FUNC      0000000000000000 |_vtable
__vtable_start 00000000002019A0 t          FUNC      0000000000000000 |_vtable
__vtable_end 00000000002019C0 t          FUNC      0000000000000000 |_vtable
__vtable_start 00000000002019E0 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201A00 t          FUNC      0000000000000000 |_vtable
__vtable_start 0000000000201A20 t          FUNC      0000000000000000 |_vtable
__vtable_end 0000000000201A40 t          FUNC      0000000000000000 |_vtable
__vtable_start 00000
```

The screenshot displays the Visual Studio Code interface with a C program in the main editor and its assembly output in the terminal.

**Source Code (q1\_313628364.c):**

```

1  #include <stdio.h>
2
3  static int square(int x) /* 3. text */
4  {
5      int result; /* 4. stack */
6      result = x * x;
7  }
8
9  int main()
10 {
11     printf("Square of %d is %d\n", 5, square(5));
12     return 0;
13 }

```

**Assembly Output (Terminal):**

Name	Value	Class	Type	Size	Line	Section
__bss_start	000000000000201024	B	NOTYPE			__bss
Completed.7698	000000000000201040	b	OBJECT	000000000000000001		__bss
__data_start	000000000000201000	D	W			__data
data_start	000000000000201000	D	W			__data
deregister_tm_clones	000000000000000500	t	FUNC			__text
doCall	000000000000000600	t	FUNC	000000000000000005f		__text
do global dtors aux	000000000000000600	t	FUNC			__text
do global dtors aux fini array entry	000000000000200d08	t	OBJECT			__fini_array
do handle	000000000000200d00	d	OBJECT			__data
DYNAMIC	000000000000200d00	d	OBJECT			__dynamic
edata	000000000000201024	D	NOTYPE			__data
end	000000000000201024	D	NOTYPE			__bss
exit@GLIBC 2.2.5	000000000000000700	U	FUNC			__UND*
fini	000000000000000700	t	FUNC			__fini
frame dummy	000000000000000000	t	FUNC			__text
frame dummy init array entry	000000000000200d00	t	OBJECT			__init_array
FRAME_END	000000000000000974	r	OBJECT			__eh_frame
GLOBAL_OFFSET_TABLE	000000000000200f00	d	OBJECT			__got
globalBuf	000000000000000c5000	B	OBJECT	000000000000010000		__bss
gmon_start	000000000000000700	r	NOTYPE			__UND*
GNU EH FRAME HDR	000000000000000528	r	FUNC			__eh_frame_hdr
init	000000000000200d08	t	NOTYPE			__init
init array end	000000000000200d08	t	NOTYPE			__init_array
init array start	000000000000200d00	t	NOTYPE			__init_array
io stdin used	000000000000000700	R	OBJECT	00000000000000000004		__rodata
ITM deregisterTMClosureTable		w	NOTYPE			__UND*
ITM registerTMClosureTable		w	NOTYPE			__UND*
Key.2775	000000000000201020	d	OBJECT	00000000000000000004		__data
libc csu fini	000000000000000700	t	FUNC	00000000000000000002		__text
libc csu init	000000000000000700	t	FUNC	00000000000000000005		__text
libc start main@GLIBC 2.2.5		U	FUNC			__UND*
main	0000000000000006f0	t	FUNC	00000000000000000025		__text
main.2776	000000000000201000	B	OBJECT	00000000000000000000		__bss
primes	000000000000201010	B	OBJECT	00000000000000000010		__data
printf@GLIBC 2.2.5		U	FUNC			__UND*
register_tm_clones	0000000000000005f0	t	FUNC			__text
square	000000000000000600	t	FUNC	00000000000000000016		__text
start	000000000000000500	t	FUNC	00000000000000000020		__text
TMC_END	000000000000201028	D	OBJECT			__data

The screenshot shows the Visual Studio Code editor with the file `q1_313628364.c` open. The code defines a `doCalc` function that calculates the cube of a number and prints it. The `main` function calls `doCalc(10)`. The terminal window shows the execution output, which is a large number of zeros followed by a square symbol, indicating a successful calculation.

```

1  #include <stdio.h>
2
3  int doCalc(int n)
4  {
5      t = val * val * val;
6      printf("The cube of %d is %d\n", val, t);
7  }
8
9  int main(int argc, char* argv[])
10 {
11     static int key = 9973;
12     static char mbuf[1024000];
13     char* p;
14
15     doCalc(key);
16 }

```

The terminal window shows the execution output, which is a large number of zeros followed by a square symbol, indicating a successful calculation.

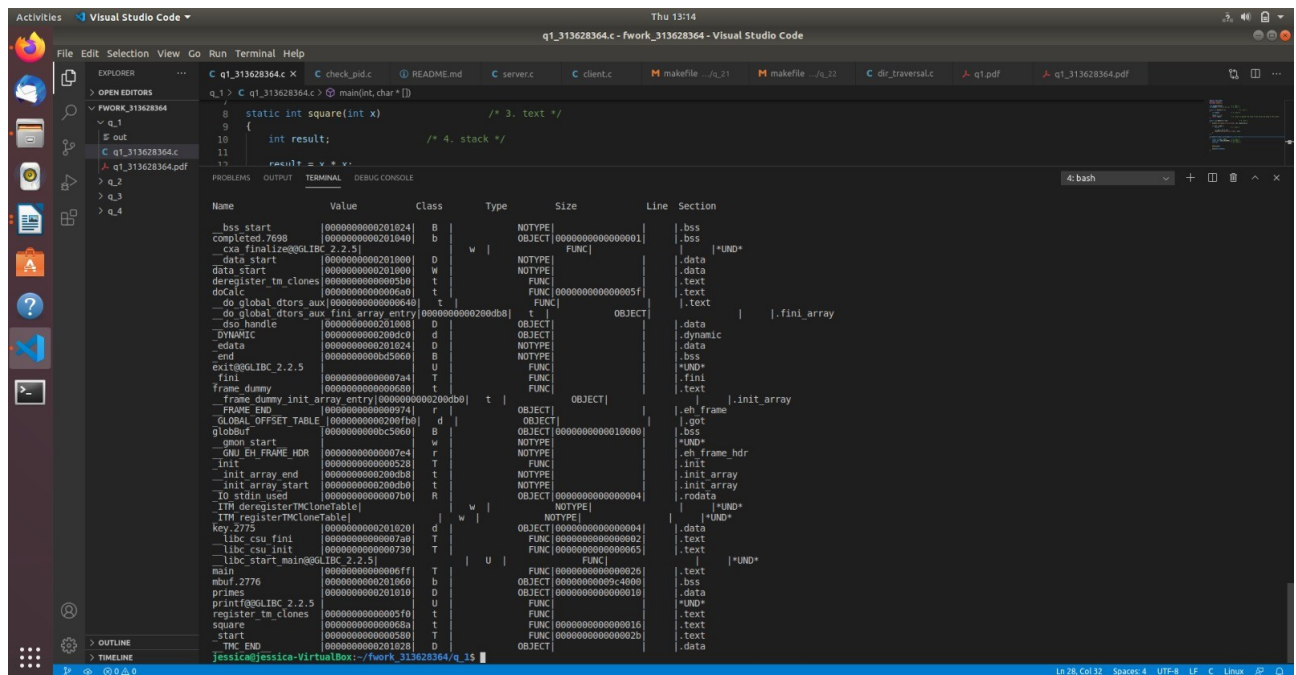
```

678:  f3 c3                rezb retq
67a:  66 0f 1f 44 00 00     nopw 0x0(%rax,%rax,1)
680:  55                    push %rbp
681:  48 89 e5              mov %rsp,%rbp
684:  5d                    pop %rbp
685:  e9 66 ff ff ff       jmpq 5f0 <register_tm_clones>
68a:  55                    push %rbp
68b:  48 89 e5              mov %rsp,%rbp
68c:  89 7d ec              mov %edi,%r14(%rbp)
691:  8b 45 ec              mov -0x14(%rbp),%eax
694:  0f af 45 ec           imul -0x14(%rbp),%eax
698:  89 45 fc              mov %eax,-0x4(%rbp)
69b:  8b 45 fc              mov -0x4(%rbp),%eax
69e:  5d                    pop %rbp
69f:  c3                   retq
0000000000000060a <doCalc>:
6a0:  55                    push %rbp
6a1:  48 89 e5              mov %rsp,%rbp
6a4:  48 83 ec 20           sub $0x20,%rsp
6a8:  89 7d ec              mov %edi,%r14(%rbp)
6ab:  8b 45 ec              mov -0x14(%rbp),%eax
6ae:  89 c7                mov %eax,%edi
6b0:  e9 d5 ff ff ff       callq 68a <square>
6b5:  89 c2                mov %eax,%edx
6b7:  8b 45 ec              mov -0x14(%rbp),%eax
6bc:  89 c5                mov %eax,%esi
6bd:  48 8d 3d f1 00 00 00  lca 0xf1(%rip),%rdi
# 764 < 10 stdin_used+0x4>

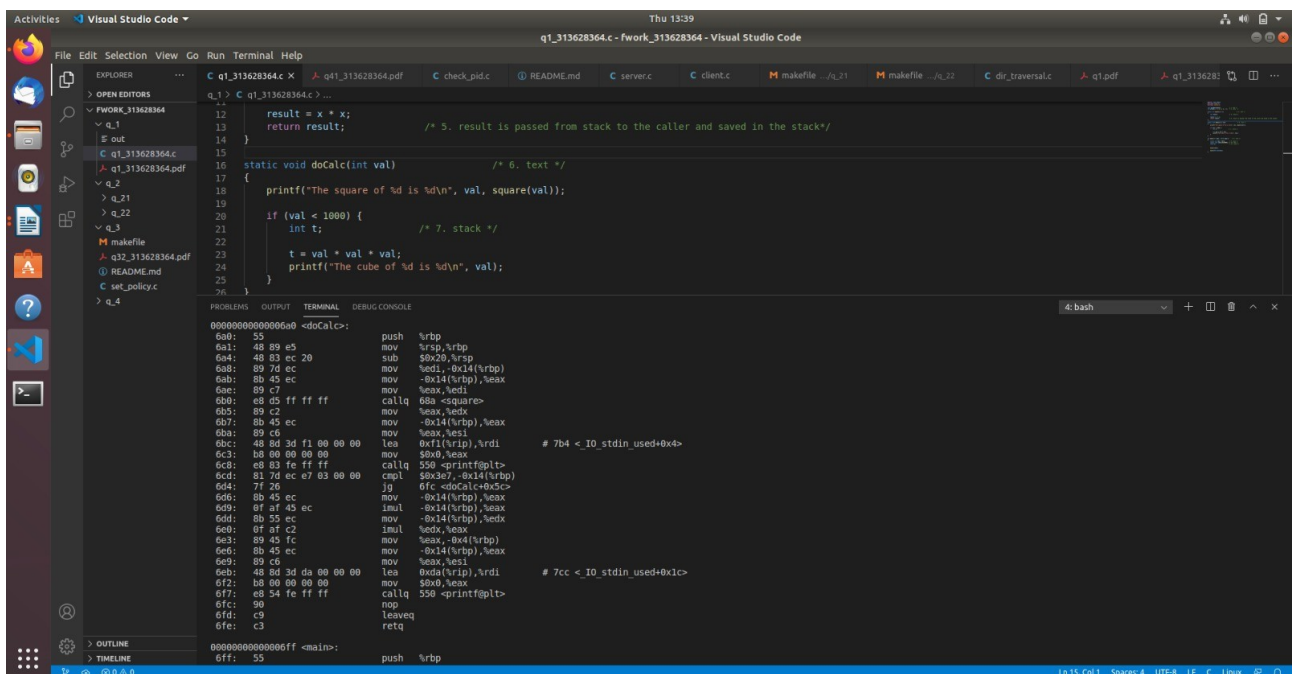
```

5. Line 13 (**return result**) – result is passed from the memory stack of the square function to the caller's stack and saves it in their stack. In this example the caller is the main function so the variable will be saved in mains stack.

6. Line 16 (**static void doCalc(int val)**) – stored in text just like all the other functions. I used the nm command like in question 3.



7. Line 21 (**int t;**) - stored in stack just like all variables in a function (non global or static variables meaning local variables). In the photo is shows that the doCalc function uses push meaning it enters variables to that functions stock.



8. Line 28 (**int main(int argc, char\* argv[])**)– All functions are saved in text as well as a main function because it is treated as any other function in the program. I used nm command to show that it is stored in text.

```

Name                                Value                                Class  Type  Size  Line  Section
-----                                -
.bss start                          0000000000201024                     B      NOTYPE 0000000000000001 .bss
completed.7088                      0000000000201040                     b      NOTYPE 0000000000000001 .bss
--cxa_finalize@GLIBC_2.2.5          0000000000201000                     D      NOTYPE 0000000000000001 .data
data start                          0000000000201000                     D      NOTYPE 0000000000000001 .data
deregister_tm_clones                0000000000200950                     t      NOTYPE 0000000000000001 .text
doCalc                              00000000002006a0                     t      FUNC   0000000000000005f .text
--do_global_ctors_aux@FINI_ARRAY_ENTRY 0000000000200c40                     t      FUNC   0000000000000005f .text
--do_global_ctors_aux@FINI_ARRAY_ENTRY 0000000000200c80                     t      FUNC   0000000000000005f .text
dso handle                          0000000000201000                     d      OBJECT 0000000000000001 .fini_array
--DYNAMIC                           00000000002009c0                     d      OBJECT 0000000000000001 .dynamic
edata                              0000000000201024                     D      NOTYPE 0000000000000001 .data
end                                0000000000200500                     B      NOTYPE 0000000000000001 .bss
exit@GLIBC_2.2.5                    0000000000200740                     U      FUNC   0000000000000001 .fini
fini                                0000000000200600                     t      FUNC   0000000000000001 .text
frame dummy                         0000000000200600                     t      OBJECT 0000000000000001 .init_array
--frame_dummy_init_array_entry@FINI_ARRAY_ENTRY 0000000000200c00                     t      OBJECT 0000000000000001 .init_array
--FRAME_END                         0000000000200974                     r      OBJECT 0000000000000001 .eh_frame
GLOBAL_OFFSET_TABLE                0000000000200f00                     d      OBJECT 0000000000000001 .got
globbuf                             0000000000200500                     b      NOTYPE 0000000000000001 .bss
--gmon_start__                     00000000002007e4                     r      NOTYPE 0000000000000001 .eh_frame_hdr
GNU EH_FRAME_HDR                   0000000000200528                     t      FUNC   0000000000000001 .init
--init_array_end                   0000000000200c80                     t      NOTYPE 0000000000000001 .init_array
--init_array_start                 0000000000200c00                     t      NOTYPE 0000000000000001 .init_array
--IO_stdin_used                     0000000000200700                     R      OBJECT 0000000000000004 .rodata
TIM deregisterTMClockTable          0000000000200730                     w      NOTYPE 0000000000000001 .text
--TIM_deregisterTMClockTable        0000000000200730                     w      NOTYPE 0000000000000001 .text
key.2775                            0000000000201020                     d      OBJECT 0000000000000004 .data
lib_csu_fini                        0000000000200720                     t      FUNC   0000000000000002 .text
lib_csu_init                        0000000000200730                     t      FUNC   0000000000000005f .text
main                                00000000002006ff                     t      U      FUNC   0000000000000026 .text
mbuf.2776                           0000000000201000                     b      OBJECT 0000000000000009:4000 .bss
primes                              0000000000201010                     D      OBJECT 0000000000000010 .data
register_tm_clones                  00000000002005f0                     t      FUNC   0000000000000010 .text
square                             0000000000200600                     t      FUNC   0000000000000010 .text
start                              0000000000200500                     t      FUNC   000000000000002b .text
TMC_END                            0000000000201028                     D      OBJECT 000000000000002b .data
jessica@jessica-VirtualBox:~/work_313628364/q_15

```

9. Line 30 (**static int key = 9973**) – stored in data segment because it is a static variable that is initialized. I used the size function to see that the stored data goes down when i delete the variable key from the program proving it is stored in data.

```

q_1.c
27
28 int main(int argc, char* argv[]) /* 8. text */
29 {
30     static int key = 9973; /* 9. data */
31     static char mbuf[1024000]; /* 10. BSS */
32     char* p; /* 11. stack */
33
34
35     doCalc(key);
36
37     exit(EXIT_SUCCESS);
38 }

```

```

jessica@jessica-VirtualBox:~/work_313628364/q_15 size out
text  data  bss  dec  hex filename
1829  628 10305568 10306196 249909 out
jessica@jessica-VirtualBox:~/work_313628364/q_15 size a.out
text  data  bss  dec  hex filename
1813  624 10305568 10306195 249905 a.out
jessica@jessica-VirtualBox:~/work_313628364/q_15

```



The screenshot displays the Visual Studio Code interface with a C program open in the editor and its disassembly view visible at the bottom.

**Editor View:**

```

q1_1.c | C q1_313628364.c | check_pid.c | README.md | server.c | client.c | makefile | q1_21 | makefile | q1_22 | dir_traversal.c | q1.pdf | q1_313628364.pdf
q1_1.c | C q1_313628364.c | main(int, char**)
8 static int square(int x) /* 3. text */
9 {
10     int result; /* 4. stack */
11
12     result = x * x;
13 }
  
```

**Disassembly View:**

Name	Value	Class	Type	Size	Line	Section
.bss_start	0000000000201824	B	NOTYPE			.bss
completed_7699	0000000000201840	B	OBJECT	0000000000000001		.bss
exa\$finalizer@GLIBC 2.2.5		w	FUNC			*UND*
.data_start	0000000000201000	D	NOTYPE			.data
data_start	0000000000201000	D	NOTYPE			.data
deregister_tm_clones	0000000000000000	t	FUNC			.text
doCall	0000000000000000	t	FUNC	0000000000000005		.text
do_global_dtors_aux	0000000000000000	t	FUNC			.text
do_global_dtors_aux_fini_array_entry	0000000000200d80	t	OBJECT			.fini_array
do_handle	0000000000201008	D	OBJECT			.data
DYNAMIC	0000000000200900	D	OBJECT			.dynamic
edata	0000000000201824	D	NOTYPE			.data
end	00000000000b0560	B	NOTYPE			.bss
exit@GLIBC 2.2.5		U	FUNC			*UND*
fini	0000000000000744	T	FUNC			.fini
frame_dummy	0000000000000000	T	FUNC			.text
frame_dummy_init_array_entry	0000000000200db0	t	OBJECT			.init_array
FRAME_END	0000000000000974	r	OBJECT			.eh_frame
GLOBAL_OFFSET_TABLE	0000000000200f60	d	OBJECT			.got
globlinit	00000000000b0560	B	OBJECT	0000000000001000		.bss
gmon_start		w	NOTYPE			*UND*
GOT_EH_FRAME_HDR	0000000000000744	r	NOTYPE			.eh_frame_hdr
.init	0000000000000000	t	FUNC			.init
.init_array_end	0000000000200d80	t	NOTYPE			.init_array
.init_array_start	0000000000200d80	t	NOTYPE			.init_array
.j0\$stdin\$used	0000000000000700	B	OBJECT	0000000000000004		.rodata
__IMD_deregisterTMCloneTable		w	NOTYPE			*UND*
__IMD_registerTMCloneTable		w	NOTYPE			*UND*
key\$2\$75	0000000000201020	d	OBJECT	0000000000000004		.data
libc_csu_fini	0000000000000740	T	FUNC	0000000000000002		.text
libc_csu_init	0000000000000730	T	FUNC	0000000000000005		.text
libc_start_main@GLIBC 2.2.5		U	FUNC			*UND*
main	000000000000096f	T	FUNC	0000000000000026		.text
scanf.2776	0000000000201060	B	OBJECT	0000000000000000		.bss
primes	0000000000201010	D	OBJECT	0000000000000018		.data
printf@GLIBC 2.2.5		U	FUNC			*UND*
register_tm_clones	0000000000000050	t	FUNC			.text
square	0000000000000060	t	FUNC	0000000000000016		.text
start	0000000000000950	T	FUNC	000000000000002b		.text
__TMC_END	0000000000201020	D	OBJECT			.data

The status bar at the bottom indicates the file path: `Ln 28, Col 32 Spaces:4 UTF-8 LF C Linux`.