```c
/* mem.c -- illustrate memory addresses, layout of C program */

#include <stdio.h>
#include <stdlib.h>

double pi = 3.14;

void fun1(int, int, int, int, int, int, int, int, int, int);
void fun2(int);
void fun3(int);

int main()
{
  int a, b, c[257], *p;
  p = malloc(sizeof(int));

  printf("starting main()\n");
  printf("  address of pi          %8x\n", &pi);
  printf("  address of a           %8x\n", &a);
  printf("  address of b           %8x\n", &b);
  printf("  address of c[0]        %8x\n", &c[0]);
  printf("  address of c[256]      %8x\n", &c[256]);
  printf("  address of pointer p   %8x\n", &p);
  printf("  address value in p     %8x\n", p);
  printf("  address of fun1        %8x\n", (unsigned) fun1);
  printf("  address of fun2        %8x\n", (unsigned) fun2);
  printf("  address of fun3        %8x\n", (unsigned) fun3);

  fun1(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);
  return 0;
}

void fun1(int a, int b, int c, int d, int e, int f, int g, int h, int i, int j)
{
  int *p;
  p = malloc(sizeof(int));

  printf("starting fun1()\n");
  printf("  address of param a     %8x\n", &a);
  printf("  address of param j     %8x\n", &j);
  printf("  address of pointer p   %8x\n", &p);
  printf("  address value in p     %8x\n", p);

  fun2(a+b+c+d+e+f+g+h+i+j);
}

void fun2(int n)
{
  int a[65536], *p;
  p = calloc(65536, sizeof(int));

  printf("starting fun2()\n");
  printf("  address of param n     %8x\n", &n);
  printf("  address of local a[0]  %8x\n", &a[0]);
  printf("  address of a[65535]    %8x\n", &a[65635]);
  printf("  address of pointer p   %8x\n", &p);
  printf("  address value in p     %8x\n", p);
  printf("  address of p[65535]    %8x\n", &p[65535]);

  fun3(16);
}

void fun3(int n)
{
  int a[16], *p;
  p = calloc(n, sizeof(int));

  printf("starting fun3()\n");
  printf("  address of param n     %8x\n", &n);
  printf("  address of local a[0]  %8x\n", &a[0]);
  printf("  address of pointer p   %8x\n", &p);
  printf("  address value in p     %8x\n", p);
}

/* output...
starting main()
  address of pi           601050
  address of a           b5ad7abc
  address of b           b5ad7ab8
  address of c[0]        b5ad76b0
  address of c[256]      b5ad7ab0
  address of pointer p   b5ad76a8
  address value in p      132c010
  address of fun1          40076a
  address of fun2          40082f
  address of fun3          40090e
starting fun1()
  address of param a     b5ad765c
  address of param j     b5ad7698
  address of pointer p   b5ad7668
  address value in p      132c030
starting fun2()
  address of param n     b5a9761c
  address of local a[0]  b5a97630
  address of a[65535]    b5ad77bc
  address of pointer p   b5a97628
  address value in p     fc7d6010
  address of p[65535]    fc81600c
starting fun3()
  address of param n     b5a975ac
  address of local a[0]  b5a975c0
  address of pointer p   b5a975b8
  address value in p      132c050
*/
```

```c
/* mem2.c - Infinite recursion to illustrate run-time stack. */

#include <stdio.h>

void fun1(int a)
{
  printf("fun1():  a = %6d, &a = %8x\n", a, &a);

  fun1(a+1);
}

int main()
{
  int x;

  printf("main():  &x = %8x\n", &x);
  printf("  address of fun1 is %8x\n", (unsigned) fun1);
  x = 1;
  fun1(x);
}

/* output...
main():  &x = d7103a1c
  address of fun1 is   40052d
fun1():  a =      1, &a = d71039fc
fun1():  a =      2, &a = d71039dc
fun1():  a =      3, &a = d71039bc
fun1():  a =      4, &a = d710399c
  ... (many lines omitted) ...
fun1():  a = 261786, &a = d69066dc
fun1():  a = 261787, &a = d69066bc
fun1():  a = 261788, &a = d690669c
fun1():  a = 261789, &a = d690667c
fun1():  a = 261790, &a = d690665c
fun1():  a = 261791, &a = d690663c
fun1():  a = 2
Segmentation fault (core dumped)
*/

```

```c
/* mem3.c - Use malloc to look at addresses on the heap.
 * Through experimentation, I found that the program aborts shortly
 * after the 200 millionth integer pointer request.  Exact results vary.
*/

#include <stdio.h>
#include <stdlib.h>

int main()
{
  int i, *p, *q;

  p = &i;
  q = malloc(sizeof(int));

  printf("address of i          %8x\n", &i);
  printf("address of pointer p  %8x\n", &p);
  printf("address of pointer q  %8x\n", &q);
  printf("address value in p    %8x\n", p);
  printf("address value in q    %8x\n", q);
  printf("\n");

  for (i = 1; i <= 210000000; ++i)
  {
    q = malloc(sizeof(int));

    if (i < 5 || i > 203000000)
      printf("i = %9d, address in q = %8x\n", i, q);
  }

  return 0;
}

/* output...
address of i          472c9eec
address of pointer p  472c9ee0
address of pointer q  472c9ed8
address value in p    472c9eec
address value in q     1dd6010

i =         1, address in q =  1dd6030
i =         2, address in q =  1dd6050
i =         3, address in q =  1dd6070
i =         4, address in q =  1dd6090
i = 203260242, address in q = 858d8a50
i = 203260243, address in q = 858d8a70
i = 203260244, address in q = 858d8a90
i = 203260245, address in q = 858d8ab0
i = 203260246, addres
Killed
*/
```