### Struct address syntax

The following questions refer to the code on the right. You may assume that (a) all required #include statements are included, (b) there are no syntax errors, (c) there are no bugs that would prevent the code from running normally. Also, you may assume that sizeof(char) == 1, sizeof(int)==4, sizeof(void*) == 8, and sizeof(double) == 8.

1) Rewrite the following expressions without using the -> operator.

<table>
<thead>
<tr>
<th>expression</th>
<th>rewritten without -&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>a_ari -&gt; ale</td>
<td></td>
</tr>
<tr>
<td>*a_bev -&gt; bug</td>
<td></td>
</tr>
<tr>
<td>&amp;a_bev -&gt; bug</td>
<td></td>
</tr>
<tr>
<td>a_bev -&gt; bug + 2</td>
<td></td>
</tr>
</tbody>
</table>

### Exploring struct objects in memory

2) Underline every local variable. How many did you find?

3) Double underline every expression that is on the data segment. How many did you find?

4) Fill in the blanks of the gdb session below.

```plaintext
(gdb) b 34
Breakpoint 1 at 0x40055b: file a.c, line 34.
(gdb) r
Starting program: /.../a
Breakpoint 1, main (argc=1, argv=0x7fffffffda78) at a.c:34
34 } 
(gdb) p sizeof(argc)
$1 = 4
(gdb) &argc 0x7fffffffda93c
(gdb) x/4bx &argc 0x7fffffffda93c: 0x01 0x00 0x00 0x00
(gdb) x/4bd &argc 0x7fffffffda93c: 1 0 0 0
```

```c
#pragma pack(1)

typedef struct {
    int ale;
    int amp;
} Ant;

typedef struct {
    double bag;
    char bib;
    char* bug;
} Bat;

int main(int argc, char** argv) {
    Ant ari = {
        .ale = 9,
        .amp = 7
    };
    Ant* a_ari = &ari;

    Bat bev = {
        .bag = 3.7,
        .bib = 'q',
        .bug = "bro"
    };
    Bat* a_bev = &bev;

    int ira[3] = {3, 4};
    return EXIT_SUCCESS;
}
```
(gdb) p ari
$2 = \{ale = 9, amp = 7\}
(gdb) p sizeof(ari)
$3 = 8
(gdb) x/4bx &ari
0x7fffffffd970: 0x09 0x00 0x00 0x00
(gdb) x/4bd a_ari
0x7fffffffd970: 0x09 0x00 0x00 0x00
(gdb) x/8bx a_ari
0x7fffffffd970: 0x09 0x00 0x00 0x00
(gdb) x/8bx ira
0x7fffffffd940: 0x09 0x00 0x00 0x00
(gdb) x/2wd ira
0x7fffffffd940: 0x09 0x00 0x00 0x00
(gdb) p bev
$4 = \{bag = 3.7000000000000002, bib = 113 'q', bug = 0x400650 "bro"\}
(gdb) x/1gx &a_be
0x7fffffffd959: 0x09 0x00 0x00 0x00
(gdb) p sizeof(bev)
$5 = 17
(gdb) p bev.bug
$6 = 0x400650 "bro"
(gdb) p &bev.bug[0]
$7 = 0x400650 "bro"
(gdb) p &bev.bug[1]
$8 = 0x400650 "bro"
(gdb) p a_ira
$9 = \{9, 7\}
(gdb) x/2wd a_ari
0x7fffffffd970: 9 7
(gdb) p a_ari
$10 = (Ant *) 0x7fffffffd970

```c
#pragma pack(1)
typedef struct {
    int ale;
    int amp;
} Ant;

typedef struct {
    double bag;
    char   bib;
    char*  bug;
} Bat;

int main(int argc, char** argv) {
    Ant ari = {
        .ale = 9,
        .amp = 7
    };
    Ant* a_ari = &ari;

    Bat bev = {
        .bag = 3.7,
        .bib = 'q',
        .bug = "bro"
    };
    Bat* a_be = &bef;

    int ira[3] = {3, 4};
    return EXIT_SUCCESS;
}"
```
In-class exercise for ECE 26400, Advanced C Programming, Prof. Alexander J. Quinn

```c
#pragma pack(1)
typedef struct {
  int ale;
  int amp;
} Ant;

typedef struct {
  double bag;
  char bib;
  char* bug;
} Bat;

int main(int argc, char** argv) {
  Ant ari = {
    .ale = 9,
    .amp = 7
  };
  Ant* a_ari = &ari;

  Bat bev = {
    .bag = 3.7,
    .bib = 'q',
    .bug = "bro"
  };
  Bat* a_bev = &bef;

  int ira[3] = {3, 4};
  return EXIT_SUCCESS;
}
```