The following is the state of memory at addresses 0x61626324 through 0x6162637f. In each line, I show:

- The pointer address to the first of four bytes.
- The four bytes in decimal.
- The four bytes in hexadecimal.
- The four bytes as chars.

The machine is little endian, but I've printed the characters in big-endian (e.g. 0, 1, 2, 3). The pointers are four bytes.

Pointer Address	Value Decimal	Value Hex	As chars +0	+1	+2	+3
0x61626324	1633837950	0x6162637e	'~'	'c'	'b'	'a'
0x61626328	1633837914	0x6162635a	'Z'	'c'	'b'	'a'
0x6162632c	1633837878	0x61626336	'6'	'c'	'b'	'a'
0x61626330	1633837916	0x6162635c	--	'c'	'b'	'a'
0x61626334	1633837951	0x6162637f	--	'c'	'b'	'a'
0x61626338	1633837931	0x6162636b	'k'	'c'	'b'	'a'
0x6162633c	1633837911	0x61626357	'W'	'c'	'b'	'a'
0x61626340	1633837920	0x61626360	'`'	'c'	'b'	'a'
0x61626344	1633837877	0x61626335	'5'	'c'	'b'	'a'
0x61626348	1633837910	0x61626356	'V'	'c'	'b'	'a'
0x6162634c	1633837905	0x61626351	'Q'	'c'	'b'	'a'
0x61626350	1633837936	0x61626370	'p'	'c'	'b'	'a'
0x61626354	1633837893	0x61626345	'E'	'c'	'b'	'a'
0x61626358	1633837873	0x61626331	'1'	'c'	'b'	'a'
0x6162635c	1633837942	0x61626376	'v'	'c'	'b'	'a'
0x61626360	1633837860	0x61626324	'$'	'c'	'b'	'a'
0x61626364	1633837905	0x61626351	'Q'	'c'	'b'	'a'
0x61626368	1633837917	0x6162635d	']'	'c'	'b'	'a'
0x6162636c	1633837883	0x6162633b	';'	'c'	'b'	'a'
0x61626370	1633837920	0x61626360	'`'	'c'	'b'	'a'
0x61626374	1633837900	0x6162634c	'L'	'c'	'b'	'a'
0x61626378	1633837944	0x61626378	'x'	'c'	'b'	'a'
0x6162637c	1633837931	0x6162636b	'k'	'c'	'b'	'a'

When you run the following procedure, the value of **p** is `0x61626324`.

```
typedef struct ms {
  char a, b, c, d;
  int e;
  char *f;
  struct ms *g;
} Mystruct;

void pm(Mystruct *p)
{
   Mystruct *q;

   printf("%c\n", p->b);
   printf("%c%c%c%c\n", p->f[0], p->f[1], p->f[2], p->f[3]);
   printf("%d\n", p[1].e);
   q = p+4;
   printf("0x%x\n", (unsigned int) q->f);
   q = p->g;
   printf("%c\n", (unsigned int) q->a);
   q = q[1].g;
   printf("%c\n", (unsigned int) q->a);
}
```

- **Question 1**: What is the first line of output?
- **Question 2**: What is the second line of output?
- **Question 3**: What is the third line of output?
- **Question 4**: What is the fourth line of output?
- **Question 5**: What is the fifth line of output?
- **Question 6**: What is the sixth line of output?

Hint #1: Your first question should be, "How many bytes are in a **Mystruct**?

Hint #2: Your second question should be, "If I have a pointer to a **Mystruct**, then where do I find the various fields, **a**, **b**, **c**, etc.?"

Hint #3: Your third question should be, "If I have a pointer **p**, to a **Mystruct**, then what is the pointer value of **p+x**?"

Answers to the clicker questions

The **Mystruct** struct has 16 bytes:

- Bytes 0 through 3: the characters **a** through **d**.
- Bytes 4 through 7: the integer **e**.
- Bytes 8 through 11: the **(char *) f**.
- Bytes 12 through 15: the **(Mystruct *) a**.

Question 1 Since **p** is `0x61626324`, **p->b** is the second of the four characters that start at that line. It is the character `'c'`.

Question 2 Since **p** is `0x61626324`, **p->f** is the pointer stored at `0x6162632c`, whose value is `0x61626336`. The four characters will the the ones that start at that adddress, so 'b', 'a', 'k' and 'c'.

Question 3 Since **p** is `0x61626324` and the **Mystruct** is 16 bytes, **p+1** will be `0x61626334`. Therefore, **p[1].e** will be the four bytes at address `0x61626338`: 1633837931.

Question 4 Since **p** is `0x61626324` and the **Mystruct** is 16 bytes, **p+4** will be `0x61626364`. Therefore, **q->f** will be the four bytes at address `0x6162636c`: 0x6162633b.

Question 5 Since **p** is `0x61626324`, we know that **p->q** is the pointer at address `0x61626330`, which has the value `0x6162635c`. So, **q** is `0x6162635c`, and **q->a** is the first character there: 'v'

Question 6 q is `0x6162635c`, so **q+1** is `0x6162636c`. That means **q[1].g** is the four bytes at `0x61626378`, which happen to also equal `0x61626378`. When we set **q** to `0x61626378`, **q->a** is the first character there: 'x'.

So the program prints:

```
c
bakc
1633837931
0x6162633b
v
x
```