ABC by Example: Struct (Top-Down Approach)

… more examples.

```
@ <stdio.hdr>

struct Foo
{
    str: array[12] of char;
    val: int;
};

fn main()
{
    local foo: Foo = {"Michael", 42};

    printf("foo.str = %s\n", foo.str);
    printf("foo.val = %d\n", foo.val);
}
```

```
MCL:tmp lehn$ abc ex2.abc
MCL:tmp lehn$ ./a.out
foo.str = Michael
foo.val = 42
```

```
@ <stdio.hdr>

struct Foo
{
    str: array[12] of char;
    val: int;
};

fn main()
{
    local foo: array[3] of Foo = {
            {"Marius", 123},
            {"Constantin", 321},
            {"Michael", 42},
        };

    for (local i: int = 0; i < sizeof(foo) / sizeof(foo[0]); ++i) {
        printf("foo[%d].str = %s\n", i, foo[i].str);
        printf("foo[%d].val = %d\n", i, foo[i].val);
    }
}
```

```
MCL:tmp lehn$ abc ex2.abc
MCL:tmp lehn$ ./a.out
foo[0].str = Marius
foo[0].val = 123
foo[1].str = Constantin
foo[1].val = 321
foo[2].str = Michael
foo[2].val = 42
```

```
@ <stdio.hdr>

struct Foo
{
    str: array[12] of char;
    val: int;
};

fn main()
{
    local foo: Foo;

    foo = (Foo){"Michael", 42};

    printf("foo.str = %s\n", foo.str);
    printf("foo.val = %d\n", foo.val);
}
```

```
MCL:tmp lehn$ abc ex2.abc
MCL:tmp lehn$ ./a.out
foo.str = Michael
foo.val = 42
```

```
@ <stdio.hdr>

struct Foo
{
    str: array[12] of char;
    val: int;
};

fn main()
{
    local foo: Foo;

    foo.str = "Michael";
    foo.val = 42;

    printf("foo.str = %s\n", foo.str);
    printf("foo.val = %d\n", foo.val);
}
```

```
MCL:tmp lehn$ abc ex2.abc

    foo.str = "Michael";
              ^^^^^^^^^^
ex2.abc:13.15-13.24: error: can not convert
an expression of  type 'array [8] of u8' to
type 'array [12] of char'
```

```
@ <stdio.hdr>

struct Foo
{
    str: array[12] of char;
    val: int;
};

fn main()
{
    local foo: Foo;

    foo.str = (array[12] of char)"Michael";
    foo.val = 42;

    printf("foo.str = %s\n", foo.str);
    printf("foo.val = %d\n", foo.val);
}
```

```
MCL:tmp lehn$ abc ex2.abc
MCL:tmp lehn$ ./a.out
foo.str = Michael
foo.val = 42
```

```
@ <stdio.hdr>

type String11: array[12] of char;

struct Foo
{
    str: String11;
    val: int;
};

fn main()
{
    local foo: Foo;

    foo.str = (String11)"Michael";
    foo.val = 42;

    printf("foo.str = %s\n", foo.str);
    printf("foo.val = %d\n", foo.val);
}
```

```
MCL:tmp lehn$ abc ex2.abc
MCL:tmp lehn$ ./a.out
foo.str = Michael
foo.val = 42
```

```
@ <stdio.hdr>

type String11: array[12] of char;

struct Foo
{
    str: String11;
    val: int;
};

fn init(x: Foo)
{
    x.str = (String11)"Michael";
    x.val = 42;
}

fn main()
{
    local foo: Foo;
    init(foo);
    printf("foo.str = %s\n", foo.str);
    printf("foo.val = %d\n", foo.val);
}
```

```
MCL:tmp lehn$ abc ex2.abc
MCL:tmp lehn$ ./a.out
foo.str = ?D???
foo.val = 32759
```

```
@ <stdio.hdr>

type String11: array[12] of char;

struct Foo
{
    str: String11;
    val: int;
};

fn init(x: Foo)
{
    x.str = (String11)"Michael";
    x.val = 42;
}

fn main()
{
    local foo: Foo = {};
    init(foo);
    printf("foo.str = %s\n", foo.str);
    printf("foo.val = %d\n", foo.val);
}
```

```
MCL:tmp lehn$ abc ex2.abc
MCL:tmp lehn$ ./a.out
foo.str =
foo.val = 0
```

```
@ <stdio.hdr>

type String11: array[12] of char;

struct Foo
{
    str: String11;
    val: int;
};

fn init(x: -> Foo)
{
    (*x).str = (String11)"Michael";
    (*x).val = 42;
}

fn main()
{
    local foo: Foo = {};
    init(&foo);
    printf("foo.str = %s\n", foo.str);
    printf("foo.val = %d\n", foo.val);
}
```

```
MCL:tmp lehn$ abc ex2.abc
MCL:tmp lehn$ ./a.out
foo.str = Michael
foo.val = 42
```

```
@ <stdio.hdr>

type String11: array[12] of char;

struct Foo
{
    str: String11;
    val: int;
};

fn init(x: -> Foo)
{
    x->str = (String11)"Michael";
    x->val = 42;
}

fn main()
{
    local foo: Foo = {};
    init(&foo);
    printf("foo.str = %s\n", foo.str);
    printf("foo.val = %d\n", foo.val);
}
```

```
MCL:tmp lehn$ abc ex2.abc
MCL:tmp lehn$ ./a.out
foo.str = Michael
foo.val = 42
```