Struct

```c
// example to illustrate C structure

#include <stdio.h>
#include <string.h>

struct customer {
    char name[10];
    int age;
    int intern_id;
};

main()
{
struct customer joe;
struct customer sue;

    joe.name[0] = 'j';
    joe.name[1] = 'o';
    joe.name[2] = 'e';
    joe.name[3] = '\0';
    joe.age = 19;
    joe.intern_id = 1200;

    printf("%s\n",joe.name);
    printf("%d\n",joe.age);
    printf("%d\n",joe.intern_id);

    strcpy(sue.name,"sue");
    sue.age = 20;
    sue.intern_id = 50;

    printf("%s\n",sue.name);
```

"main1.c" 36L, 562C

Handwritten annotations:

ignore alignment

→ 10 bytes
→ 4 bytes
→ 4 bytes
} 18 byte

} use 4 bytes only not 10 bytes

Ready

```
        joe.name[0] = 'j';
        joe.name[1] = 'o';
        joe.name[2] = 'e';
        joe.name[3] = '\0';
        joe.age = 19;
        joe.intern_id = 1200;

        printf("%s\n",joe.name);
        printf("%d\n",joe.age);
        printf("%d\n",joe.intern_id);

        strcpy(sue.name,"sue");
        sue.age = 20;
        sue.intern_id = 50;

        printf("%s\n",sue.name);
        printf("%d\n",sue.age);
        printf("%d\n",sue.intern_id);

}
"main1.c" 36L, 562C written
pod1-1 61 % gcc main1.c
main1.c:12:1: warning: return type defaults to 'int
 main()
 ^
pod1-1 62 % a.out
joe
19
1200
sue
20
50
pod1-1 63 %
```

struct customer listA [100];

typedef

```
// example to illustrate C structure and typedef

#include <stdio.h>

typedef struct customer {
    char name[10];
    int age;
    int intern_id;
} customer_t;

main()
{
customer_t joe;

        joe.name[0] = 'j';
        joe.name[1] = 'o';
        joe.name[2] = 'e';
        joe.name[3] = '\0';
        joe.age = 19;
        joe.intern_id = 1200;

        printf("%s\n",joe.name);
        printf("%d\n",joe.age);
        printf("%d\n",joe.intern_id);

}


"main2.c" 26L, 396C
```

→ **shortcut**

→ **don't need to use**
   struct customer

```
// example to illustrate C structure and member malloc()

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

typedef struct customer {
    char *name;
    int age;
    int intern_id;
} customer_t;

main()
{
customer_t joe;

    joe.name = (char *) malloc(10*sizeof(char));
    strcpy(joe.name,"joe");

    joe.age = 19;
    joe.intern_id = 1200;

    printf("%s\n",joe.name);
    printf("%d\n",joe.age);
    printf("%d\n",joe.intern_id);
```

main4. c

```
// example to illustrate C structure and member malloc()

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

typedef struct customer {
    char name[10];
    int age;
    int intern_id;
} customer_t;

main()
{
customer_t *joe;

    joe = (customer_t *) malloc(sizeof(customer_t));
    joe->name[0] = 'j';
    joe->name[1] = 'o';
    joe->name[2] = 'e';
    joe->name[3] = '\0';

    joe->age = 19;
    joe->intern_id = 1200;

    printf("%s\n",joe->name);
    printf("%d\n",joe->age);
"main4.c" [converted] 30L, 487C                    14,1
```

Ready   ssh2: AES-256-CTR   14, 1   27 Rows, 72 Cols   VT100

* joe;   syntax different

joe →name

joe → age

```
} customer_t;

main()
{
customer_t *joe;

    joe = (customer_t *) malloc(sizeof(customer_t));
    joe->name[0] = 'j';
    joe->name[1] = 'o';
    joe->name[2] = 'e';
    joe->name[3] = '\0';

    joe->age = 19;
    joe->intern_id = 1200;

    printf("%s\n",joe->name);
    printf("%d\n",joe->age);
    printf("%d\n",joe->intern_id);

}

sslab01 24 % gcc main4.c
sslab01 25 % a.out
joe
19
1200
sslab01 26 %
```

Ready   ssh2: AES-256-CTR   27, 14   27 Rows, 72 Cols   V

File   Edit   View   Options   Transfer   Script   Tools   Wind

Enter host <Alt+R>

ssiab01.cs.purdue.edu

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

typedef int my_t;
typedef my_t my1_t;
typedef my1_t my2_t;

main()
{
my2_t x;

x = 3;
   printf("%d\n",x);

}
```

"main5.c" [converted] 15L, 151C

print : 3

(joe → (mira → name)) ← anothe struct in ⟨ joe struct ⟩

(joe . mira . name) in stack