C Programming
Day17.B

2017.11.11

structure

alias

new type

```c
#include <stdio.h>

typedef unsigned int UINT;


int main(void) {

  unsigned int u1 = 0xffffffff;
  UINT         u2 = 0xffffffff;

  printf("sizeof(unsigned int) = %ld \n", sizeof
(unsigned int));
  printf("sizeof(UINT) = %ld \n", sizeof(UINT));

  if (u1>0) printf("u1 > 0 \n");
  if (u2>0) printf("u2 > 0 \n");

}
```

```c
#define NEW 333
```

Can use NEW
instead of 333

Find-replace in an editor

Struct Variable Declaration (1)

structure type

```
struct aaa {
        int     i;
        short   s;
        char    c;
} ;
```

definition

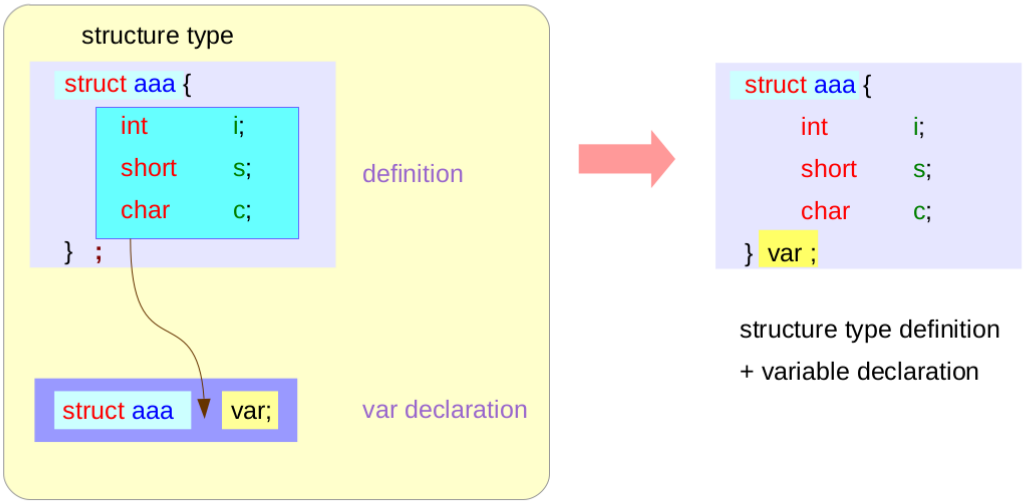
```
struct aaa  var;
```

var declaration

structure type

```
struct aaa {
        int     i;
        short   s;
        char    c;
} var ;
```

Struct Variable Declaration (2)



Struct Variable Declaration (3)

structure type

```
struct aaa {
        int     i;
        short   s;
        char    c;
} ;
```

```
typedef struct aaa  ATYPE ;
```

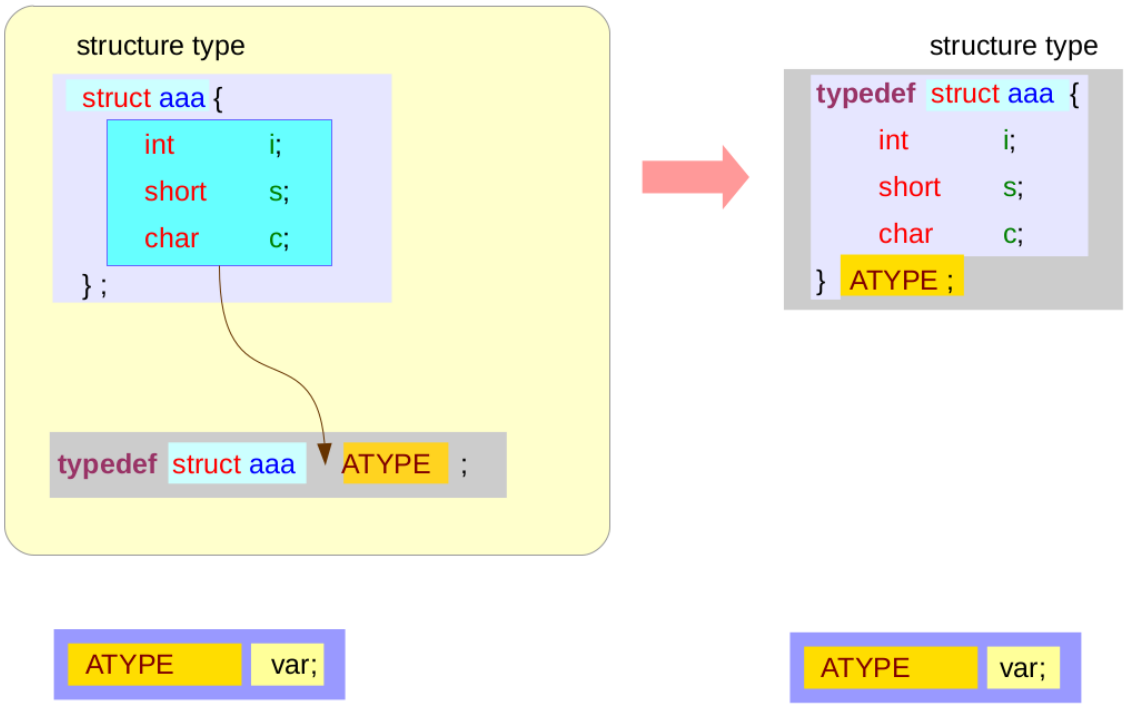
```
ATYPE   var;
```

structure type

```
typedef struct aaa {
        int     i;
        short   s;
        char    c;
} ATYPE ;
```

```
ATYPE   var;
```

Struct Variable Declaration (4)



Struct Variable Declaration Summary

structure type

```
struct aaa {
        int     i;
        short   s;
        char    c;
} ;
```

```
struct aaa  var;
```

structure type

```
struct aaa {
        int     i;
        short   s;
        char    c;
} ;
```

```
typedef struct aaa  ATYPE ;
```

```
ATYPE  var;
```

structure type

```
struct aaa {
        int     i;
        short   s;
        char    c;
} var ;
```

structure type

```
typedef struct aaa {
        int     i;
        short   s;
        char    c;
} ATYPE ;
```

```
ATYPE  var;
```

Structure Variable Declaration

structure type

```
struct aaa {
        int     i;
        short   s;
        char    c;
} ;
```

definition

```
struct aaa  var;
```

var declaration

accessing a structure variable

&var

&var = &var.i

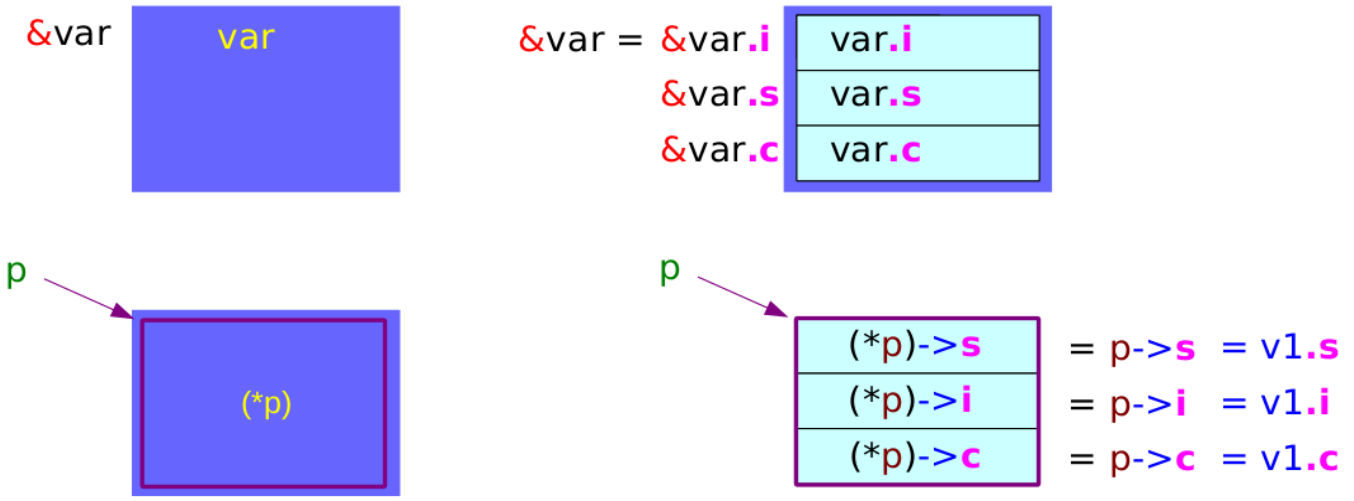
&var.s

&var.c

accessing its members

Dereferencing Pointers to Structures (1)

(*p).mem ⟺ p->mem



```c
#include <stdio.h>

int main(void) {

   struct aaa {
     int    i;  // 4-byte
     short  s;  // 2-byte
     char   c;  // 1-byte
   } ;

   struct aaa X;

   X.i = 300;  // int   variable : X.i
   X.s = 200;  // short variable : X.s
   X.c = 100;  // char  variable : X.c

   printf("X.i= %d \n", X.i);
   printf("X.s= %d \n", X.s);
   printf("X.c= %d \n", X.c);

}
```

← write
← write
← write

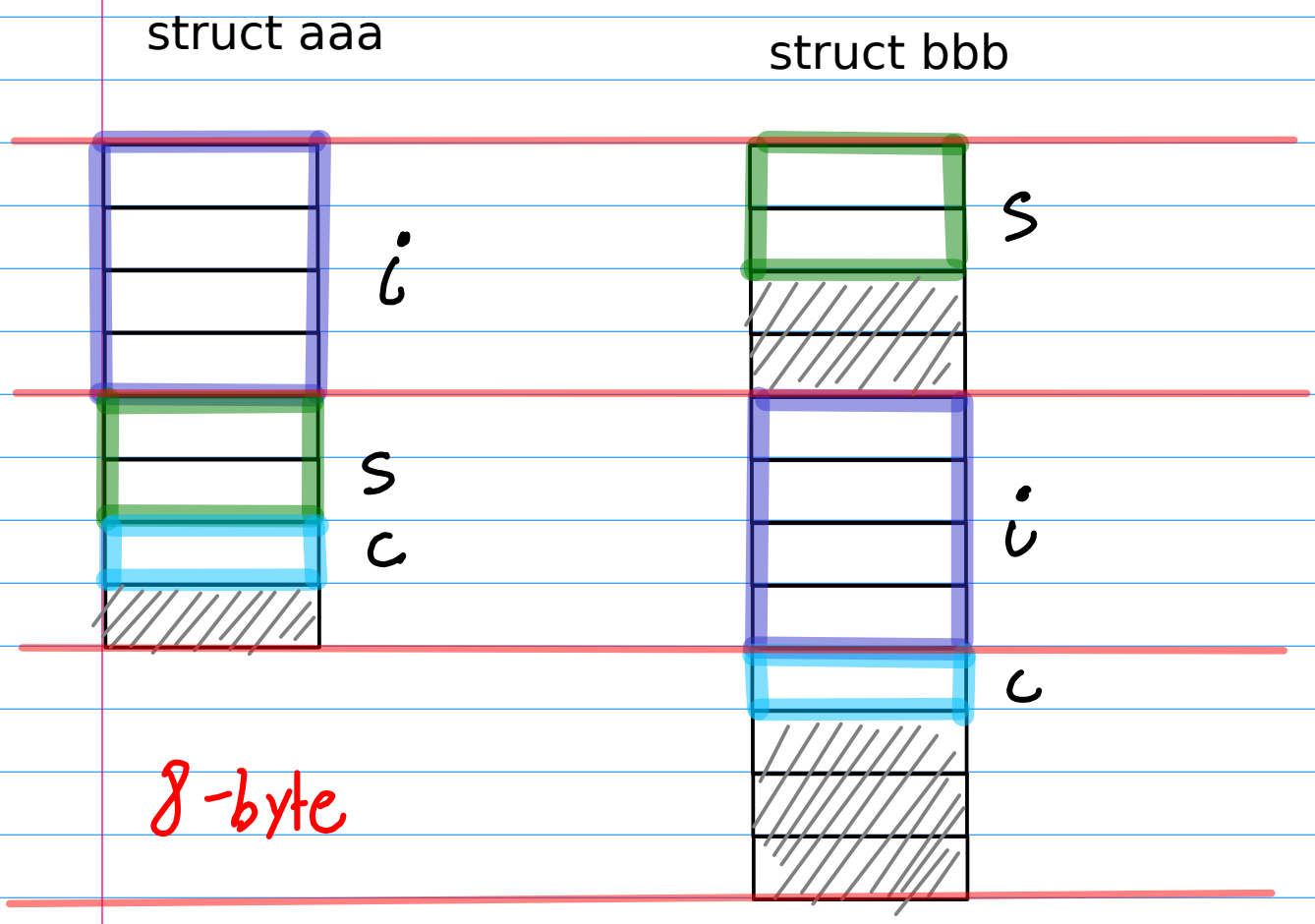
→ Read
→ Read
→ Read

X.i an int variable
X.s a short variable
X.c a char variable

```c
#include <stdio.h>

struct aaa {
  int    i;  // 4-byte
  short  s;  // 2-byte
  char   c;  // 1-byte
} ;


struct bbb {
  short  s;  // 2-byte
  int    i;  // 4-byte
  char   c;  // 1-byte
} ;


int main(void) {
   struct aaa  X = { 11, 22, 33 };
   struct bbb  Y = { 10, 20, 30 };

   printf("sizeof(X)  = %ld \n", sizeof(X));
   printf("sizeof(X.i)= %ld \n", sizeof(X.i));
   printf("sizeof(X.s)= %ld \n", sizeof(X.s));
   printf("sizeof(X.c)= %ld \n", sizeof(X.c));

   // printf("X  = %p \n", X  ); // Not Working
   printf("X.i= %d \n", X.i);
   printf("X.s= %d \n", X.s);
   printf("X.c= %d \n", X.c);

   printf("&X  = %p \n", &X);
   printf("&X.i= %p \n", &X.i);
   printf("&X.s= %p \n", &X.s);
   printf("&X.c= %p \n", &X.c);


   printf("sizeof(Y)  = %ld \n", sizeof(Y));
   printf("sizeof(Y.s)= %ld \n", sizeof(Y.s));
   printf("sizeof(Y.i)= %ld \n", sizeof(Y.i));
   printf("sizeof(Y.c)= %ld \n", sizeof(Y.c));

   // printf("Y  = %p \n", Y  ); // Not Working
   printf("Y.s= %d \n", Y.s);
   printf("Y.i= %d \n", Y.i);
   printf("Y.c= %d \n", Y.c);

   printf("&Y  = %p \n", &Y);
   printf("&Y.s= %p \n", &Y.s);
   printf("&Y.i= %p \n", &Y.i);
   printf("&Y.c= %p \n", &Y.c);


}
```

initializers

4-byte alignment




```
sizeof(X)  = 8
sizeof(X.i)= 4
sizeof(X.s)= 2
sizeof(X.c)= 1
X.i= 11
X.s= 22
X.c= 33
&X   = 0x7fff335a4de0
&X.i= 0x7fff335a4de0
&X.s= 0x7fff335a4de4
&X.c= 0x7fff335a4de6
sizeof(Y)  = 12
sizeof(Y.s)= 2
sizeof(Y.i)= 4
sizeof(Y.c)= 1
Y.s= 10
Y.i= 20
Y.c= 30
&Y   = 0x7fff335a4df0
&Y.s= 0x7fff335a4df0
&Y.i= 0x7fff335a4df4
&Y.c= 0x7fff335a4df8
```

```c
#include <stdio.h>

struct aaa {
  int    i;  // 4-byte
  short  s;  // 2-byte
  char   c;  // 1-byte
} ;


struct bbb {
  int    i;  // 4-byte
  short  s;  // 2-byte
  char   c;  // 1-byte
} ;


int main(void) {
   struct aaa  X = { 11, 22, 33 };
   struct aaa  Y = { 10, 20, 30 };
   struct bbb  Z = { 10, 20, 30 };

   printf("-------------\n");
   printf("X.i= %d \n", X.i);
   printf("X.s= %d \n", X.s);
   printf("X.c= %d \n", X.c);

   printf("-------------\n");
   printf("Y.i= %d \n", Y.i);
   printf("Y.s= %d \n", Y.s);
   printf("Y.c= %d \n", Y.c);

   Y = X;

   printf("-------------\n");
   printf("Y.i= %d \n", Y.i);
   printf("Y.s= %d \n", Y.s);
   printf("Y.c= %d \n", Y.c);

   // if (X == Y) printf("the same \n");  // Not Working

   // Y = Z; // Not Working

   // Y = (struct aaa) Z; // Not Working

   Y.i = Z.i;
   Y.s = Z.s;
   Y.c = Z.c;
}
```

```
-------------
X.i= 11
X.s= 22
X.c= 33
-------------
Y.i= 10
Y.s= 20
Y.c= 30
-------------
Y.i= 11
Y.s= 22
Y.c= 33
```

the only operation allowed

struct aaa ≠ struct bbb

```c
#include <stdio.h>

struct aaa {
  int    i;  // 4-byte
  short  s;  // 2-byte
  char   c;  // 1-byte
} ;


int main(void) {
   struct aaa  X = { 100, 200, 300 };
   struct aaa *P;

   P = &X;

   printf("sizeof(P)  = %ld \n", sizeof(P));
   printf("sizeof(X)  = %ld \n", sizeof(X));
   printf("sizeof(X.i)= %ld \n", sizeof(X.i));
   printf("sizeof(X.s)= %ld \n", sizeof(X.s));
   printf("sizeof(X.c)= %ld \n", sizeof(X.c));

   printf("X.i= %d \n", X.i);
   printf("X.s= %d \n", X.s);
   printf("X.c= %d \n", X.c);

   printf("&X  = %p \n", &X);
   printf("&X.i= %p \n", &X.i);
   printf("&X.s= %p \n", &X.s);
   printf("&X.c= %p \n", &X.c);

   printf("&P      = %p \n", &P);
   printf(" P      = %p \n",  P);
   printf("(*P).i  = %d \n", (*P).i);
   printf("(*P).s  = %d \n", (*P).s);
   printf("(*P).c  = %d \n", (*P).c);
   printf("P->i  = %d \n", P->i);
   printf("P->s  = %d \n", P->s);
   printf("P->c  = %d \n", P->c);


}
```

0 ~ 255

300 - 256 = 44

```
sizeof(P)  = 8
sizeof(X)  = 8
sizeof(X.i)= 4
sizeof(X.s)= 2
sizeof(X.c)= 1
X.i= 100
X.s= 200
X.c= 44
&X  = 0x7fff6ca460a0
&X.i= 0x7fff6ca460a0
&X.s= 0x7fff6ca460a4
&X.c= 0x7fff6ca460a6
&P      = 0x7fff6ca460b0
 P      = 0x7fff6ca460a0
(*P).i  = 100
(*P).s  = 200
(*P).c  = 44
P->i  = 100
P->s  = 200
P->c  = 44
```

```c
#include <stdio.h>

struct aaa {
  int    i;  // 4-byte
  short  s;  // 2-byte
  char   c;  // 1-byte
} ;


int main(void) {
   struct aaa  X;
   struct aaa *P;

   X.i = 300;  // int   variable : X.i
   X.s = 200;  // short variable : X.s
   X.c = 100;  // char  variable : X.c

   P = &X;


   P->i = 0x7FFFFFFF;
   P->s = 0x7FFF;
   P->c = 0x7F;

   printf("X.i= %d \n", X.i);
   printf("X.s= %d \n", X.s);
   printf("X.c= %d \n", X.c);

   printf("X.i= %#13x \n", X.i);
   printf("X.s= %#13x \n", X.s);
   printf("X.c= %#13x \n", X.c);
}
```

```
X.i= 2147483647
X.s= 32767
X.c= 127
X.i=    0x7fffffff
X.s=         0x7fff
X.c=           0x7f
```