Introduction to Scientific and Engineering Computation (BIL 104E)

Lab 12

Structures and Function Calls

In the C language, it is possible to pass an entire structure to a function. In addition, the return value of the function can be a structure.

```
#include <stdio.h>
   struct computer {
         float cost;
         int year;
         int cpu_speed;
         char cpu_type[16];
   };
  typedef struct computer SC; /* create synonym */
 SC DataReceive(SC s); /* function declaration */
main(void)
{
   SC model;
   model = DataReceive(model);
   printf("Here are what you entered:\n");
   printf("Year: %d\n", model.year);
   printf("Cost: $%6.2f\n", model.cost);
   printf("CPU type: %s\n", model.cpu_type);
   printf("CPU speed: %d MHz\n", model.cpu_speed);
```

Structures and Function Calls

```
    getchar();
    getchar();
    return 0;
}
/* function definition */
SC DataReceive(SC s)
{
    printf("The type of the CPU inside your computer?\n");
    gets(s.cpu_type);
    printf("The speed(MHz) of the CPU?\n");
    scanf("%d", &s.cpu_speed);
    printf("The year your computer was made?\n");
    scanf("%d", &s.year);
    printf("How much you paid for the computer?\n");
    scanf("%f", &s.cost);
    return s;
}
```

Structures and Function Calls

Computer Screen

```
The type of the CPU inside your computer?
core 4
The speed(MHz) of the CPU?
3200
The year your computer was made?
2013
How much you paid for the computer?
1255.35
Here are what you entered:
Year: 2013
Cost: $1255.35
CPU type: core 4
CPU speed: 3200 MHz
```

Referencing Structures with Pointers

```
#include <stdio.h>
struct computer {
    float cost;
    int year;
    int cpu_speed;
    char cpu_type[16];
};

typedef struct computer SC;
void DataReceive(SC *ptr_s);
```

Referencing Structures with Pointers

```
main(void)
{
   SC model;
   DataReceive(&model);
   printf("Here are what you entered:\n");
   printf("Year: %d\n", model.year);
   printf("Cost: $%6.2f\n", model.cost);
   printf("CPU type: %s\n", model.cpu_type);
   printf("CPU speed: %d MHz\n", model.cpu_speed);
   getchar();
   getchar();
   return 0;
}
void DataReceive(SC *ptr_s)   /* function definition */
{
   printf("The type of the CPU inside your computer?\n");
   gets((*ptr_s).cpu_type);
   printf("The speed(MHz) of the CPU?\n");
   scanf("%d", &(*ptr_s).cpu_speed);
   printf("The year your computer was made?\n");
   scanf("%d", &(*ptr_s).year);
   printf("How much you paid for the computer?\n");
   scanf("%f", &(*ptr_s).cost);
}
```

Referencing Structures with Pointers

Computer Screen

```
The type of the CPU inside your computer?
core 4
The speed(MHz) of the CPU?
3500
The year your computer was made?
2013
How much you paid for the computer?
1560
Here are what you entered:
Year: 2013
Cost: $1560.00
CPU type: core 4
CPU speed: 3500 MHz
```

Arrays of Structures

```
#include<stdio.h>
struct class_list{
      char *name;
      char *sur_name;
      int number;
      int midterm;
      int final;
      int semester;
};
typedef struct class_list student;
main()
{
     student bil104[5];
     student *ptr_s;
     int i;

     ptr_s = bil104;

     ptr_s -> name = "Murat";
     ptr_s -> sur_name = "Simsek";
     ptr_s -> number = 110012094;
     ptr_s -> midterm = 70;
     ptr_s -> final = 80;
     (bil104[0]).semester = ((bil104[0].midterm) + (bil104[0].final))/2 ;
```

Arrays of Structures

```
    (bil104[1]).name = "Hakan";
    (bil104[1]).sur_name = "Kuzu";
    (bil104[1]).number = 110012012;
    (bil104[1]).midterm = 50;
    (bil104[1]).final = 75;
    (bil104[1]).semester = ((ptr_s -> midterm) + (ptr_s -> final))/2 ;

    for (i = 0; i < 2; i++) {
        printf("%d %s  %s  %d\n",ptr_s ->number,ptr_s->name,ptr_s ->sur_name, ptr_s -> semester);
       ++ptr_s;
    }

    getchar();
    return 0;
}
```

Computer Screen

```
110012094    Murat      Simsek    75
110012012    Hakan      Kuzu    62
```

Nested Structures

```
#include<stdio.h>
struct exam_result {
	int quiz;
	int midterm;
	int final;
};
typedef struct exam_result exams;
struct class_list {
	char *name;
	char *surname;
	int number;
	exams bil104;
};
typedef struct class_list C_class;

main()
{
	C_class student[4] = {{"Murat", "Simsek", 110012025, {50, 70, 80} },
				{"Salih", "Yorulmaz", 110012045, {20, 30, 50}},
				{"Ugur", "Ince", 110012090, {10, 20, 60}}};

	C_class *ptr_std;
	int i;
	double semester;
```

This syntax is valid only for initialization

Nested Structures

```
    student[3].name = "Servet";
    student[3].surname = "Gok";
    student[3].number = 110009144;
    student[3].bil104.quiz = 0;
    student[3].bil104.midterm = 30;
    student[3].bil104.final = 50;

    ptr_std = student;
    printf("Number    Name   Surname   Semester\n");
    for(i = 0; i < 4; i++){
        semester = ptr_std -> bil104.quiz * 0.3 + ptr_std -> bil104.midterm * 0.4 + ptr_std -> bil104.quiz * 0.4;
        printf("%d %s  %s  %2.2lf\n", ptr_std -> number, ptr_std -> name, ptr_std -> surname, semester  );
        ++ptr_std;
    }

    getchar();
    return 0;
}
```

This syntax is valid only for assignment

```
Number    Name   Surname   Semester
110012025 Murat  Simsek  63.00
110012045 Salih  Yorulmaz  26.00
110012090 Ugur  Ince  15.00
110009144 Servet  Gok  12.00
```

Initializing a Union

```
main()
{
   union employee {
            int start_year;
            int dpt_code;
            int id_number;
   } info;
   info.year = 1997;
   info.dpt_code = 8;
   info.id_number = 1234;
   printf("%d    %d     %d", info.year , info.dpt_code, info.id_number);
}
```

Computer Screen

```
1234    1234    1234
```

Initializing a Union

```
#include<stdio.h>
main()
{
   union   u {
           char ch;
           int x;
   }  a_union;

   char *ptr_ch;
   int i;

   ptr_ch = &(a_union.ch);
   printf("address of  member ch = %p\n", ptr_ch);
   printf("integer member is changed\n" );
   a_union.x = 365;
   printf("ch = %c\n", a_union.ch );
   printf("x = %d\n", a_union.x );
   for(i = 1; i < 5; i++)
        printf("%d byte = %d\n",i, *(ptr_ch++));
```

256 + 109 = 365

0	0	0	0	0	0	0	1	0	1	1	0	1	1	0	1

'm'

Memory Address		
0022FF44	109	1-byte
0022FF45	1	1-byte
0022FF46	0	1-byte
0022FF47	0	1-byte

Initializing a Union

256 + 72 = 328

0	0	0	0	0	0	0	1

0	1	1	0	1	1	0	1

'H'

```
    printf("char member is changed\n");
    a_union.ch = 'H';
    printf("ch = %c\n", a_union.ch );
    printf("x = %d\n", a_union.x );
    ptr_ch = &(a_union.ch);

    for(i = 1; i < 5; i++)
        printf("%d byte = %d\n",i, *(ptr_ch++));

    printf("H = %d\n", 'H');
    printf("m = %d\n", 'm');

    getchar();
    return 0;
}
```

Memory Address		
0022FF44	72	1-byte
0022FF45	1	1-byte
0022FF46	0	1-byte
0022FF47	0	1-byte

Initializing a Union

Computer Screen

```
address of  member ch = 0022FF44
integer member is changed
ch = m
x = 365
1 byte = 109
2 byte = 1
3 byte = 0
4 byte = 0
char member is changed
ch = H
x = 328
1 byte = 72
2 byte = 1
3 byte = 0
4 byte = 0
H = 72
m = 109
```

The Size of a Union

```
#include <stdio.h>
#include <string.h>
main(void)
{
   union u {
      double x;
      int y;
   } a_union;

   struct s {
      double x;
      int y;
   } a_struct;

   printf("The size of double: %d-byte\n", sizeof(double));
   printf("The size of int: %d-byte\n", sizeof(int));

   printf("The size of a_union: %d-byte\n", sizeof(a_union));
   printf("The size of a_struct: %d-byte\n", sizeof(a_struct));

   getchar();
   return 0;
}
```

Computer Screen

```
The size of double: 8-byte
The size of int: 4-byte
The size of a_union: 8-byte
The size of a_struct: 16-byte
```

Using Unions

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

struct survey {
   char name[20];
   char c_d_p;
   int age;
   union {
        char cable_company[16];
        char dish_company[16];
   } provider;
};
void DataEnter(struct survey *s);
void DataDisplay(struct survey *s);

main(void)
{
   struct survey tv;
   DataEnter(&tv);
   DataDisplay(&tv);
   getchar();
   getchar();
   return 0;
}
```

Cable_company and dish company are similar and they can be saved same memory area using union.

Using Unions

```
void DataEnter(struct survey *ptr)
{   char is_yes[4];
    printf("Are you using cable at home? (Yes or No)\n");
    gets(is_yes);
    if ((is_yes[0] == 'Y') || (is_yes[0] == 'y'))  {
       printf("Enter the cable company name:\n");
       gets(ptr->provider.cable_company);
       ptr->c_d_p = 'c';
    }
    else {
         printf("Are you using a satellite dish? (Yes or No)\n");
         gets(is_yes);
         if ((is_yes[0] == 'Y') || (is_yes[0] == 'y')){
              printf("Enter the satellite dish company name:\n");
              gets(ptr->provider.dish_company);
              ptr->c_d_p = 'd';
         }
         else
              ptr->c_d_p = 'p';
    }
    printf("Please enter your name:\n");
    gets(ptr->name);
    printf("Your age:\n");
    scanf("%d", &ptr->age);
}
```

Using Unions

```
/* function definition */
void DataDisplay(struct survey *ptr)
{
   printf("\nHere's what you've entered:\n");
   printf("Name: %s\n", ptr->name);
   printf("Age: %d\n", ptr->age);
   if (ptr->c_d_p == 'c')
      printf("Your cable company is: %s\n", ptr->provider.cable_company);
   else
       if (ptr->c_d_p == 'd')
           printf("Your satellite dish company is: %s\n", ptr->provider.dish_company);
       else
           printf("You don't have cable or a satellite dish.\n");
   printf("\nThanks and Bye!\n");
}
```

Using Unions

Computer Screen

```
Are you using cable at home? (Yes or No)
n
Are you using a satellite dish? (Yes or No)
y
Enter the satellite dish company name:
dsmart
Please enter your name:
murat
Your age:
36

Here's what you've entered:
Name: murat
Age: 36
Your satellite dish company is: dsmart

Thanks and Bye!
```

Defining Bit Fields with struct

```
#include<stdio.h>
struct bit_field {
      int gender: 1; // 1 indicates male 0 indicates female
      int condition: 1; // 1 indicates successful 0 indicates failed
};
typedef struct bit_field bf;
struct class_list {
      char *name;
      bf bit3long;
};
typedef struct class_list class;
main()
{    class bil104[3] = { {"Tolga", 1, 1}, {"Elif", 0, 0} , {"Merve", 0, 1} };
     int i;
     for(i = 1; i < 4; i++) {
          printf("%d   %s  ", i, bil104[i-1].name);
          if(bil104[i-1].bit3long.gender)
               printf("  MALE");
          else
               printf("  FEMALE");
          if(bil104[i-1].bit3long.condition)
               printf("  successful");
          else
               printf("  failed");
          printf("\n");
     }
      getchar();
      return 0;
}
```

Defining Bit Fields with struct

Computer Screen

```
1   Tolga    MALE  successful
2   Elif    FEMALE  failed
3   Merve    FEMALE  successful
```