Програмчлалын үндэс
Лекц № 9

Бүтэц, нэгдэл, тоочих төрөл

А. Хүдэр

Агуулга

- Бүтцийг үүсгэх, анхны утга олгох, хэрэглэх
- Бүтцийг функц руу утгаар нь болон хаягаар нь дамжуулах (typedef)
- Нэгдлийг үүсгэх, анхны утга олгох, хэрэглэх
- Тоочих төрлийг үүсгэх, анхны утга олгох, хэрэглэх
- Файлыг үүсгэх, унших, бичих, шинэчлэх
- Файл боловсруулах хоёр арга:
  - Дараалсан хандалт
  - Санамсаргүй хандалт

Бүтцийг зарлах

- Жишээ

```
struct card {
   char *face;
   char *suit;
 };
```

  - `struct` түлхүүр үгээр `card` бүтцийг тодорхойлно
  - `card` нь бүтцийн нэр бөгөөд энэ төрлийн хувьсагчдыг зарлахад хэрэглэгдэнэ
  - `card` нь `char *` төрлийн хоёр гишүүнтэй байна
    - эдгээр гишүүд нь `face` болон `suit` гэсэн нэртэй

Бүтцийг зарлах

- `struct`-ын тухай
  - struct нь өөрийн хуулбарыг дотроо агуулж чадахгүй
  - Ижилхэн бүтэц төрөл рүү заах гишүүнтэй байж болно
  - Бүтцийг зарлахад санах ойд зай нөөцөлдөггүй
    - Оронд нь бүтэц төрлийн хувьсагч зарлах шинэ төрлийг үүсгэдэг
- Зарлалт
  - Бусад хувьсагчтай адил зарлагдана:

    ```
    struct card oneCard, deck[52], *cPtr;
    ```

  - Таслалаар тусгаарлагдсан жагсаалт хэрэглэж болно

    ```
    struct card {
        char *face;
        char *suit;
    } oneCard, deck[52], *cPtr;
    ```

Бүтцийг зарлах

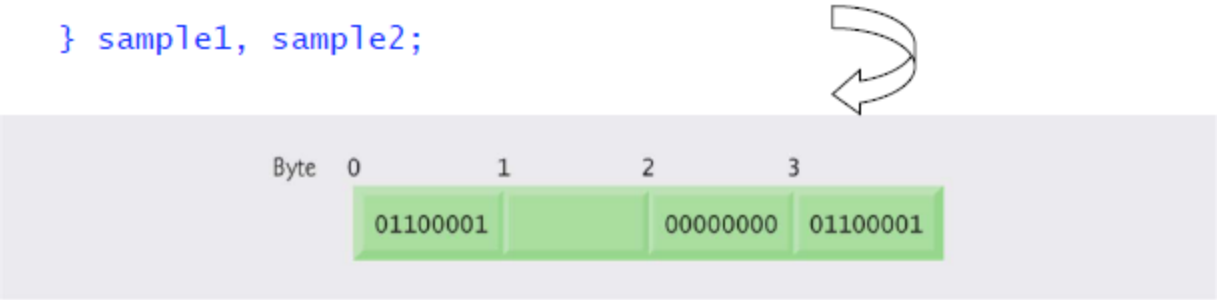
- Үйлдлүүд
  - Ижил төрлийн бүтцэд бүтцийн утга олгох
  - Бүтцийн хаягийг авах (&)
  - Бүтцийн гишүүдэд хандах
  - Бүтцийн хэмжээг тогтоохын тулд sizeof – ыг хэрэглэх

Тайлбар

- Бичлэгийн гишүүд санах ойд заавал дараалан байрлах албагүй. Компьютер зарим үед **хагас үг, үг** болон **давхар үгийг** өөр өөр газар хадгалдаг тул гишүүдийн хооронд нүх гарч болно. Үг гэдэг нь компьютерт мэдээллийг хадгалах стандарт нэгж ба 4 эсвэл 8 байт байна.

```
struct example {
    char c;
    int i;
} sample1, sample2;
```

sample1 ба sample2 бүтцүүдийн гишүүдийн утга адилхан байж болох ч дундаа байгаа нүхэн дэх утга өөр байж болох тул тэнцүү биш байна



- struct төрлийн хувьсагчийг санах ойд хадгалах үед ‘a’ утгатай тэмдэгт болон 97 утгатай бүхэл тооны хооронд нүх үүссэн байдлыг харуулав.

Бүтцэд анхны утга олгох

- Анхны утгын жагсаалт
  - Жишээ

    struct card oneCard = { "Three", "Hearts" };
- Утга олгох үйлдэл
  - Жишээ

    struct card threeHearts = oneCard;
  - threeHearts-ийг доорх байдлаар зарлан, утга олгож болно:

    struct card threeHearts;

    threeHearts.face = “Three”;

    threeHearts.suit = “Hearts”;

Бүтцийн гишүүдэд хандах

- Бүтцийн гишүүнд хандах
  - Цэг (.) үйлдлийг бүтцийн хувьсагч дээр ашиглана

    ```
    struct card myCard;
    printf( "%s", myCard.suit );
    ```

  - Сум операторыг бүтцийн заагч хувьсагч дээр ашиглана

    ```
    struct card *myCardPtr = &myCard;
    printf( "%s", myCardPtr->suit );
    ```

  - myCardPtr->suit нь доорхтой ижил

    ```
    ( *myCardPtr ).suit
    ```

```c
/* Example 87:
   Using the structure member and
   structure pointer operators */
#include <stdio.h>

/* card structure definition */
struct card {
   char *face; /* define pointer face */
   char *suit; /* define pointer suit */
}; /* end structure card */

int main( void )
{
   struct card aCard; /* define one struct card variable */
   struct card *cardPtr; /* define a pointer to a struct card */

   /* place strings into aCard */
   aCard.face = "Ace";
   aCard.suit = "Spades";
```

Бүтэц зарлах

Бүтцийн зарлалт цэгтэй таслалаар төгсөнө

Цэг оператор нь бүтцийн гишүүнд хандана

```
20
21    cardPtr = &aCard; /* assign address of aCard to cardPtr */
22
23    printf( "%s%s%s\n%s%s%s\n%s%s%s\n", aCard.face, " of ", aCard.suit,
24       cardPtr->face, " of ", cardPtr->suit,
25       ( *cardPtr ).face, " of ", ( *cardPtr ).suit );
26
27    return 0; /* indicates successful termination */
28
29 } /* end main */
```

```
Ace of Spades
Ace of Spades
Ace of Spades
```

Сум оператор нь бүтцийн заагчийн гишүүнд хандана

Агуулга

- Бүтцийг үүсгэх, анхны утга олгох, хэрэглэх
- Бүтцийг функц руу утгаар нь болон хаягаар нь дамжуулах (typedef)
- Нэгдлийг үүсгэх, анхны утга олгох, хэрэглэх
- Тоочих төрлийг үүсгэх, анхны утга олгох, хэрэглэх
- Файлыг үүсгэх, унших, бичих, шинэчлэх
- Файл боловсруулах хоёр арга:
  - Дараалсан хандалт
  - Санамсаргүй хандалт

Бүтцийг функцтэй хэрэглэх

- Бүтцийг функц руу дамжуулах
  - Бүтцийг бүхлээр нь дамжуулах
    - Эсвэл гишүүн гишүүнээр нь дамжуулах
  - Аль ч тохиолдолд утгаар дамжуулагдана
- Бүтцийг хаягаар дамжуулах
  - Хаягийг нь дамжуулах
- Хүснэгтийг утгаар дамжуулах
  - Хүснэгт төрлийн гишүүнтэй бүтэц үүсгэх
  - Бүтцийг дамжуулах

typedef

- `typedef`
  - Шинээр тодорхойлсон төрөлд өөр нэр (зохиомол нэр) өгнө
  - Төрлийн нэрийг богиносгохын тулд `typedef` -ийг хэрэглэнэ
  - Жишээ:

    typedef struct Card *CardPtr;
  - `struct Card *` төрлийн зохиомол нэр нь `CardPtr` гэсэн шинэ төрөл тодорхойлно
  - `typedef` нь шинэ төрөл үүсгэдэггүй
    - Зөвхөн өмнө нь зохиосон төрөлд зохиомол нэр өгнө

Жишээ: Хөзрийг хурдан хольж тараах загвар

- Псевдокод:
  - card бүтэц төрлийн хүснэгт үүсгэх
  - Хөзрүүдийг модонд оруулах
  - Модыг холих
  - Хөзрийг тараах

```c
/* Example 88:
   The card shuffling and dealing program using structures */
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

/* card structure definition */
struct card {
   const char *face; /* define pointer face */
   const char *suit; /* define pointer suit */
}; /* end structure card */

typedef struct card Card; /* new type name for struct card */

/* prototypes */
void fillDeck( Card * const wDeck, const char * wFace[],
   const char * wSuit[] );
void shuffle( Card * const wDeck );
void deal( const Card * const wDeck );

int main( void )
{
   Card deck[ 52 ]; /* define array of Cards */

   /* initialize array of pointers */
   const char *face[] = { "Ace", "Deuce", "Three", "Four", "Five",
      "Six", "Seven", "Eight", "Nine", "Ten",
      "Jack", "Queen", "King"};

```

`card` бүр нь дүрс болон өнгөтэй байна

`Card` нь `struct card` -ын өөр нэр болно

```
30    /* initialize array of pointers */
31    const char *suit[] = { "Hearts", "Diamonds", "Clubs", "Spades"};
32
33    srand( time( NULL ) ); /* randomize */
34
35    fillDeck( deck, face, suit ); /* load the deck with Cards */
36    shuffle( deck ); /* put Cards in random order */
37    deal( deck ); /* deal all 52 Cards */
38
39    return 0; /* indicates successful termination */
40
41 } /* end main */
42
43 /* place strings into Card structures */
44 void fillDeck( Card * const wDeck, const char * wFace[],
45    const char * wSuit[] )
46 {
47    int i; /* counter */
48
49    /* loop through wDeck */
50    for ( i = 0; i <= 51; i++ ) {
51       wDeck[ i ].face = wFace[ i % 13 ];
52       wDeck[ i ].suit = wSuit[ i / 13 ];
53    } /* end for */
54
55 } /* end function fillDeck */
56
```

`Card` төрлийн өөрчилж болох хүснэгтийн тогтмол заагч

`Card` бүрт дүрс болон өнгө өгч модуудыг үүсгэнэ

```
57 /* shuffle cards */
58 void shuffle( Card * const wDeck )
59 {
60    int i;      /* counter */
61    int j;      /* variable to hold random value between 0 - 51 */
62    Card temp; /* define temporary structure for swapping Cards */
63
64    /* loop through wDeck randomly swapping Cards */
65    for ( i = 0; i <= 51; i++ ) {
66       j = rand() % 52;
67       temp = wDeck[ i ];
68       wDeck[ i ] = wDeck[ j ];
69       wDeck[ j ] = temp;
70    } /* end for */
71
72 } /* end function shuffle */
73
74 /* deal cards */
75 void deal( const Card * const wDeck )
76 {
77    int i; /* counter */
78
79    /* loop through wDeck */
80    for ( i = 0; i <= 51; i++ ) {
81       printf( "%5s of %-8s%c", wDeck[ i ].face, wDeck[ i ].suit,
82          ( i + 1 ) % 2 ? '\t' : '\n' );
83    } /* end for */
84
85 } /* end function deal */
```

`Card` бүрийг өөр санамсаргүй карттай сольж модыг холино

```
Four of Clubs          Three of Hearts
Three of Diamonds       Three of Spades
 Four of Diamonds         Ace of Diamonds
 Nine of Hearts           Ten of Clubs
Three of Clubs           Four of Hearts
Eight of Clubs           Nine of Diamonds
Deuce of Clubs          Queen of Clubs
Seven of Clubs           Jack of Spades
  Ace of Clubs           Five of Diamonds
  Ace of Spades          Five of Clubs
Seven of Diamonds         Six of Spades
Eight of Spades         Queen of Hearts
 Five of Spades         Deuce of Diamonds
Queen of Spades           Six of Hearts
Queen of Diamonds       Seven of Hearts
 Jack of Diamonds         Nine of Spades
Eight of Hearts          Five of Hearts
 King of Spades           Six of Clubs
Eight of Diamonds         Ten of Spades
  Ace of Hearts          King of Hearts
 Four of Spades          Jack of Hearts
Deuce of Hearts          Jack of Clubs
Deuce of Spades           Ten of Diamonds
Seven of Spades          Nine of Clubs
 King of Clubs            Six of Diamonds
  Ten of Hearts          King of Diamonds
```

Агуулга

- Бүтцийг үүсгэх, анхны утга олгох, хэрэглэх
- Бүтцийг функц руу утгаар нь болон хаягаар нь дамжуулах (typedef)
- Нэгдлийг үүсгэх, анхны утга олгох, хэрэглэх
- Тоочих төрлийг үүсгэх, анхны утга олгох, хэрэглэх
- Файлыг үүсгэх, унших, бичих, шинэчлэх
- Файл боловсруулах хоёр арга:
  - Дараалсан хандалт
  - Санамсаргүй хандалт

Нэгдэл

- Нэгдэл
  - Гишүүд санах ойн нэг ижил хэсгийг эзэлжлэн ашиглана
  - Нэгдлийн гишүүн нь ямар ч төрөлтэй байж болно
  - Нэгдлийн санах ойд эзлэх хэмжээ нь түүний гишүүдийн хамгийн томыг багтаахад хангалттай байна
  - Хандах бүрд зөвхөн нэг гишүүн, нэг өгөгдлийн төрлөөр ашиглагдана
- Нэгдлийг зарлах
  - Бүтэцтэй адилхан, өөрөөр хэлбэл
    - union number {
      - int x;
      - float y;
    - };
    - union number value;

Нэгдэл

- Үйлдлүүд
  - Ижил төрлийн нэгдэлд утгыг олгох: =
  - Хаяг авах: &
  - Нэгдлийн гишүүдэд хандах: .
  - Заагч ашиглан гишүүдэд хандах: ->
- Зарлах үед нэгдлийн анхны утгыг түүний эхний гишүүний төрлөөр олгож болно, өөрөөр хэлбэл

  union number value = {10};

  union number value = {1.43}

  нь хөвөх таслалтай тооны бүхэл хэсгийг авах ба хөрвүүлэгчээс анхааруулга өгнө.

```c
/* Example 89:
   An example of a union */
#include <stdio.h>

/* number union definition */
union number {
   int x;
   double y;
}; /* end union number */

int main( void )
{
   union number value; /* define union variable */

   value.x = 100; /* put an integer into the union */
   printf( "%s\n%s\n%s\n  %d\n\n%s\n  %f\n\n\n",
      "Put a value in the integer member",
      "and print both members.",
      "int:", value.x,
      "double:", value.y );

```

Нэгдлийг зарлах

Нэгдлийн зарлалт цэгтэй таслалаар төгсөнө

y утгагүй байгааг анхаар

```
22    value.y = 100.0; /* put a double into the same union */
23    printf( "%s\n%s\n%s\n  %d\n\n%s\n  %f\n",
24       "Put a value in the floating member",
25       "and print both members.",
26       "int:", value.x,
27       "double:", value.y );
28
29    return 0; /* indicates successful termination */
30
31 } /* end main */
```

y-т утга өгснөөр x-ийн утгыг арилгана

```
Put a value in the integer member
and print both members.
int:
  100

double:
  -92559592117433136000000000000000000000000000000000000000000000000000.000000


Put a value in the floating member
and print both members.
int:
  0

double:
  100.000000
```

Агуулга

- Бүтцийг үүсгэх, анхны утга олгох, хэрэглэх
- Бүтцийг функц руу утгаар нь болон хаягаар нь дамжуулах (typedef)
- Нэгдлийг үүсгэх, анхны утга олгох, хэрэглэх
- Тоочих төрлийг үүсгэх, анхны утга олгох, хэрэглэх
- Файлыг үүсгэх, унших, бичих, шинэчлэх
- Файл боловсруулах хоёр арга:
  - Дараалсан хандалт
  - Санамсаргүй хандалт

Тоочих төрлийн тогтмолууд

- Тоочих төрөл
  - Нэр өгсөн бүхэл тоон төрлийн тогтмолууд
  - Тоочих төрлийн тогтмолууд нь тэмдэгтэн тогтмолуудтай адил ба утгуудыг нь автоматаар олгоно
    - Утга нь тэгээс эхлэн нэгээр нэмэгдэн олгогдоно
    - Утгыг зориуд = үйлдлээр олгож болно
    - Тогтмолуудын нэр нь ялгаатай байх ёстой
  - Жишээ:

    enum Months { JAN = 1, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC};
    - Доторх тогтмолууд нь 1-ээс 12 хүртэл утга авах enum Months гэсэн шинэ төрөл үүсгэж байна
  - Тоочих төрлийн хувьсагч нь тоочих төрлийн тогтмол утгуудаа л авна (бүхэл тоон хэлбэрийн утга авахгүй)

```c
/* Example 90:
   Using an enumeration type */
#include <stdio.h>

/* enumeration constants represent months of the year */
enum months {
   JAN = 1, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC };

int main( void )
{
   enum months month; /* can contain any of the 12 months */

   /* initialize array of pointers */
   const char *monthName[] = { "", "January", "February", "March",
      "April", "May", "June", "July", "August", "September", "October",
      "November", "December" };

```

Тоочих төрлөөр JAN тогтмолд 1 гэсэн утга, дараагийн тогтмолуудад 2, 3, 4, ... гэх мэт утга онооно

```
18    /* loop through months */
19    for ( month = JAN; month <= DEC; month++ ) {
20       printf( "%2d%11s\n", month, monthName[ month ] );
21    } /* end for */
22
23    return 0; /* indicates successful termination */
24 } /* end main */
```

```
 1    January
 2   February
 3      March
 4      April
 5        May
 6       June
 7       July
 8     August
 9  September
10    October
11   November
12   December
```

Тэмдэгт тогтмолын адил тоочих төрлийн тогтмолууд нь хөрвүүлэлтийн үед тоон утгаараа солигдоно