**Unions**

* Only one member at a time can be accessed by the union.
* To access and print size should be fixed

Handshakes takes place where union can be useful

Based on the availability of the server the loader is called or invoked

These helps to stream the video in smarter way

**Enumerated datatype**

* It is a set of named integers
* The integers or the values are named here
* For a whole number value a name can be given

Ex: we have employee id given even though we have names

1 #include <stdio.h>

2

3 enum daysOfWeek

4 {

5 Monday,

6 Tuesday,

7 Wednesday,

8 Thursday,

9 Friday,

10 Saturday,

11 Sunday

12 }Days;

13

14 int main()

15 {

16 Days = Thursday;

17 printf("\nDay = %d",Days);

18

19 printf("\n\n");

20 return 0;

21 }

22

~

O/P

user50@trainux01:~/practice/try/day13$ gcc enum.c

user50@trainux01:~/practice/try/day13$ ./a.out

Day = 3

* If we specify any values for the days then it will print that and takes the consecutive call

**File handling**

Start of file and end of file indicators

FILE is the naming convention which is used to change the content of the file

Two types of files:

1. Sequential file
2. Random file

To access a file we need a file

FILE \*fp;

FILE \*fopen(char\*name,char\*mode);

fp = fopen(“file name”,”mode”);

Here the cursor will blink at end of file but in vieditor it is at at the beginning

If the file is able to open then it needs to be able to return the address

If fp = NULL then we are unable to open the file as path is undefined

Refer to man fopen to get more information

1 #include <stdio.h>

2 #include <string.h>

3 #include <stdlib.h>

4

5 int main()

6 {

7 FILE \*fd = NULL;

8 fd = fopen("Test1.txt","w");

9

10 if(fd == NULL)

11 {

12 perror("fopen: ");

13 printf("\n%p\n\n",fd);

14 exit(1);

15 }

16 printf("\nOpened the file successfully %p \n\n",fd);

17

18 fclose(fd);

19 return 0;

20 }

21

O/P

user50@trainux01:~/practice/try/day13$ gcc file1.c

user50@trainux01:~/practice/try/day13$ ./a.out

Opened the file successfully 0x55558425a260

user50@trainux01:~/practice/try/day13$ ll

total 44

drwxrwxr-x 2 user50 user50 4096 Nov 12 06:39 ./

drwxrwxr-x 8 user50 user50 4096 Nov 12 04:36 ../

-rw-r--r-- 1 user50 user50 12288 Nov 12 05:14 .clrcode.c.swp

-rw-rw-r-- 1 user50 user50 0 Nov 12 06:39 Test1.txt

-rwxrwxr-x 1 user50 user50 8472 Nov 12 06:39 a.out\*

-rw-rw-r-- 1 user50 user50 0 Nov 12 05:08 clrcode.c

-rw-rw-r-- 1 user50 user50 214 Nov 12 05:07 enum.c

-rw-rw-r-- 1 user50 user50 284 Nov 12 06:39 file1.c

-rw-rw-r-- 1 user50 user50 458 Nov 12 04:57 union.c

/\*

FILE = Dsata Structure

fd ==> file pointer to FILE DS

fopen("NameofFile","Mode") ==> opening a file

Mode ==> r,w,a,r+,w+,a+

fclose(fd) ==> close the opened file

read/write

fprintf,fscanf ==> formatted writing and reading respectively

fputs,fgets ==> unformatted i/o operation

fwrite/fread ==> reading or writing binary objects

\*/

Man fgets and man fputs

Assignment

Read the contents from the database and store it in structure and print the structure

The record of employees ---

101|Amit Kumar1|M|8888|10001

102|Amit Kumar2|M|8889|10002

103|Amit Kumar3M|8810|10003

104|Amit Kumar4M|8811|10004

105|Amit Kumar5|M|8812|10005

Db.txt

#include <stdio.h>

#include <string.h>

// Define a structure to hold employee information

struct Employee {

int emp\_id;

char name[100]; // assuming the name won't be longer than 99 characters

char gender;

int salary;

int dept\_id;

};

int main() {

// Define an array to hold 5 employee records (since we have 5 records)

struct Employee employees[5];

// Input data for each employee

// Using fscanf to read data from stdin (this can be from a file or standard input)

for (int i = 0; i < 5; i++) {

// Reading and parsing the employee data

// Example input line: 101|Amit Kumar1|M|8888|10001

// We use sscanf to read the fields split by '|'

char input[200];

fgets(input, sizeof(input), stdin);

// Parsing input into structure

sscanf(input, "%d|%99[^|]|%c|%d|%d",

&employees[i].emp\_id,

employees[i].name,

&employees[i].gender,

&employees[i].salary,

&employees[i].dept\_id);

}

// Print the employee information

printf("Employee Records:\n");

for (int i = 0; i < 5; i++) {

printf("Employee ID: %d\n", employees[i].emp\_id);

printf("Name: %s\n", employees[i].name);

printf("Gender: %c\n", employees[i].gender);

printf("Salary: %d\n", employees[i].salary);

printf("Department ID: %d\n\n", employees[i].dept\_id);

}

return 0;

}

Refer other

For compilation ----------------

Gcc -o ./bin/app -I ./inc/ ./src

NOTE:

CC=gcc

cflags=-g -c

oflags=-o

OBJ=./obj

SRC=./src

INC=./inc

BIN=./bin

iflags =-I$(INC)/

$(BIN)/app:$(OBJ)/main.o $(OBJ)/emp.o

$(CC) $(oflags) $(BIN)/app $(OBJ)/main.o $(OBJ)/emp.o

$(OBJ)/main.o:$ Continued……

Allocate memory based on the number of records ----- Using linked list but can also be done using another for loop while fgets is printed

By using system we cannot execute the commands of linux

SEEK\_END, SEEK\_SET and SEEK\_CUR are used for numeric characters

Otherwise it gives error

Man on fread and fwrite

Fread(0) will not read anything so atleast we need to give as fread(1)

And gives to pointer