

# STRUCTURES

19/09/17

```
struct tag-name (optional)
{
    datatype1 member1;
    datatype2 member2;
};

e.g struct book
```

char name[20];	20
char au-name[30];	30
int pages;	2
float price;	4
	<u>56</u>
	sizeof(book)

continuous allocation of memory according to  
the order of data types declared defined!

struct tag-name var-name

```
e.g struct book b1;
```

struct

{

char name [20];

char au-name [30];

}; b1, b2, b3;

→ By default variables of this structure.

struct book b1 = { "XYZ", "abc", 100, 120.5 }, b2, b3;

b1.name

b1.au-name

b1.page

b1.price

Q. WAP to create a structure date then input date of joining of one employee and print it.

struct date

{

int day;

int month;

int year;

};

void main()

{

struct date doj;

printf("enter DOJ");

scanf("%d %d %d", &doj.day, &doj.month, &doj.year);

printf("%d %d %d DOJ DAY DOJ MONTH DOJ YEAR");

getch();

};

Q. WAP to create a structure student to store information like name, roll no. and %age of student, then input information for 2 students and print roll no. of that student whose %age more.

struct student

```
char name[20];
int roll;
float per;
```

}

void main()

{

```
struct student s1, s2;
gets(s1.name);
gets(s2.name);
scanf("%d%.f", &s1.roll, &s1.per);
scanf("%d%.f", &s2.roll, &s2.per);
```

if(s1.per > s2.per)

printf("%d", s1.roll);

else

printf("%d", s2.roll);

getch();

}

Q. Input the values for n no. of students and print name of those students whose percentage is more than 75%.

struct student

{

char name[20];

int roll;

float per;

}

void main()

{

struct student s[50];

int i, n;

scanf("%d", &n);

for(i=0; i<n; i++)

{

gets(s[i].name);

scanf("%d%.f", &s[i].roll, &s[i].per);

{

for(i=0; i<n; i++)

{

if (s[i].per > 75)

printf("%f %s", s[i].name);

{

getch();

{

Q. Create a structure student to store name, roll no. and marks in 5 subjects.  
WAP to input information for n no. of students and print %age of all students.

struct student

{

char name[20];

int roll;

float mark[5];

}

void main()

{

struct student s[100];

float avg;

int i, j, n, sum = 0;

scanf("%d", &n);

for(i=0; i<n; i++)

{ gets(s[i].name);

scanf("%d", &s[i].roll);

for(j=0; j<5; j++)

{

scanf("%f", &s[i].mark[j]);

}

}

for(i=0; i<n; i++)

{ sum = 0;

for(j=0; j<5; j++)

{

sum += s[i].mark[j];

}

avg = sum/5;

printf("%f", avg);

getch(); }

Q. Create a structure employee to store information like name, Date of Joining and salary.

WAP to input information for n no. of employees and print total salary distributed in a month.

① struct employee  
  {  
    char name[30];  
    int day;  
    int month;  
    int year;  
    float salary;  
};

void main()  
{

    struct employee e[50];  
    int i, n;  
    scanf ("%d", &n);  
    for (i=0; i<n; i++)  
    {  
        gets (e[i].name);

② struct date

  {  
    int day;  
    int month;  
    int year;  
};

struct employee

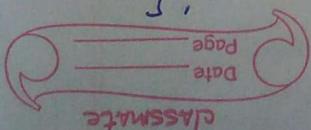
  {  
    char name[20];  
    struct date doj;  
    int salary;  
};

③ struct employee  
  {

    char name[20];  
    struct date  
    {  
        int day;  
        int month;  
        int year;  
    } doj;  
    int salary;  
};

void main

  {  
    struct employee e[50];  
    int i, sum=0, n;  
    scanf ("%d", &n);  
    for (i=0; i<n; i++)  
    {  
        gets (e[i].name);  
        scanf ("%d %d %d",  
               &e[i].doj.day,  
               &e[i].doj.month,  
               &e[i].doj.year);  
        sum += e[i].salary;  
    }  
    printf ("Total salary = %d", sum);



```

scanf("%d", e[i].salary);
}
for(i=0; i<n; i++)
{
    sum += e[i].salary;
}
printf("%d", sum);
getch();
}

```

## UNION

union tmp → size of union = highest  
 memory occupying member  
 int x;  
 float y;  
 char z;  
 ;

## BIT FIELD

decide the length of the variable; user defined

struct tmp  
 {
 int day : 1; → applied only on integer  
 data type
 }

total sum of assigned bit field can not be more than  
size of integer variable.

=> accessing members :

```
scanf("%d", temp);  
c.day = temp;
```

## FILE HANDLING

- permanent storage

text files (.txt) and binary files can be accessed through C programming.

FILE \*fp;

```
fp = fopen ("file name", "mode");
```

[ r → read  
w → write  
a → append

pre defined  
structure

char-var: getc - get a single character from the file.

putc - to write character into the file.

int-var: gets - get no. from the file.

putw - to put no. into the file.

```
getc(fp);
```

```
putc(data, fp);
```

EOF - End of file

ctrl + Z - save file

```
fclose(fp);
```

Q. WAP to input characters in a file, then read the characters from the file and count how many of them are vowels.

```
void main()
{
    int c = 0;
    FILE *fp; char ch;
    printf("enter characters");
    fp = fopen("charc.txt", "w");
    while ((ch = getchar()) != EOF)
    {
        putc(ch, fp);
    }
    fclose(fp);
    fp = fopen("charc.txt", "r");
    while (ch = getc(fp)) != EOF
    {
        ch = toupper(ch);
        if (ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U')
            c++;
    }
    fclose(fp);
    printf("%d", c);
    getch();
}
```

Q. WAP to input characters, convert it in lowercase in a file and then read the characters from the file and display in uppercase.

```
void main()
```

Date \_\_\_\_\_  
Page \_\_\_\_\_

classmate

```

the
of them

FILE * fp
printf ("enter characters");
fp = fopen ("file.txt", "w");
while (ch = getchar () != EOF)
{
    putc (ch, fp);
}
fclose (fp);
fp = fopen ("file.txt", "r");
while (ch = getc (fp)) != EOF
{
    printf ("%c", ch - 32);
}
fclose (fp);
getch();
}

```

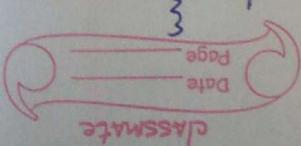
Q. WAP to input no. in a file and then count how many of them are even no..

```

void main()
{
    FILE *fp;
    int c = 0, n, num, i;

    fp = fopen ("test.txt", "w");
    printf ("Enter limit");
    scanf ("%d", &n);
    for (i = 0; i < n; i++)
    {
        scanf ("%d", &num);
        puts (num, fp);
    }
}

```



```

fclose(fp);
fp = fopen("test.txt", "r");
for (i = 0; i < n; i++)
{
    num = getw(fp);
    if (num % 2 == 0)
        c++;
}

```

```

fclose(fp);
printf("%d", c);

```

Q. WAP to input no. in a file and copy those no. in two files on the basis of +ve & -ve no.

```

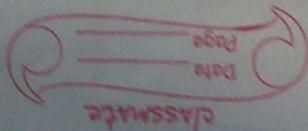
void main()
{
FILE *fp, *f1, *f2;
int n, num, i;
fp = fopen("test.txt", "r");
printf("Enter limit:");
scanf("%d", &n);
for (i = 0; i < n; i++)
{
    scanf("%d", &num);
    putw(num, fp);
}

```

```

fclose(fp);
fp = fopen("test.txt", "r");
f1 = fopen("positive.txt", "w");
f2 = fopen("negative.txt", "w");
for (i = 0; i < n; i++)
{

```



```

    num = getw(fp)
    if (num > 0)
        putw(num, f1);
    else
        putw(num, f2);
}

```

take a counter in both  
cond to use for  
loop instead.

```

fclose(fp);
fclose(f1);
rewind(fp);

```

```

fp = fopen ("test.txt", "r");
for (i=0; i<n; i++)
{

```

```

    getw(num, fp)
    printf ("%d", num);
}

```

```

fclose(fp);

```

```

f1 = fopen ("positive.txt", "r");

```

```

for (i=0; i<n; i++) while (num = getw(f1)) != EOF
{

```

```

    getw(num, f1)
    printf ("%d", num);
}

```

```

fclose(f1);

```

```

f2 = fopen ("negative.txt", "r");

```

```

for (i=0; i<n; i++) while (num = getw(f2)) != EOF
{

```

```

    getw(num, f2)
    printf ("%d", num);
}

```

```

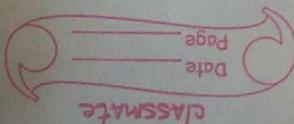
fclose(f2);

```

```

getch();
}

```



Q. WAP to input +ve integers in a file , then calculate factorial of all no. using function fact() and save the calculated factorial into another file.

```

int fact (int);
void main ()
{
    FILE *fp,*fc;
    int n, num, i, f;
    fp = fopen ("factorial.txt", "w");
    printf ("enter limit");
    scanf ("%d", &n);
    for (i=0; i<n; i++)
    {
        scanf ("%d", &num);
        putw (num, fp);
    }
    fclose (fp);
    fp=fopen ("factorial.txt", "r");
    fc=fopen ("f.txt", "w");
    for (i=0; i<n; i++)
    {
        num = getw (fp);
        if = fact (num);
        printf ("%d", if);
        putw (if, fc);
    }
    fclose (fp);
    fclose (fc);
    int fact (int num)
    {
        int f=1, i;
        for (i=2; i<=num; i++)
            f = f*i;
        return (f);
    }
}

```

