#include <stdio.h>

int main()

{

char Lines[] = "101|Amit|M|8888|10000";

char Names[10][20];

int row=0,col=0;

char \*ptr=NULL;

int flag = 0;

ptr = Lines;

while(\*ptr!='\0')

{

putchar(\*ptr);

ptr++;

}

ptr = Lines;

row=0;

col=0;

while(\*ptr != '\0')

{

if(\*ptr == '|')

break;

Names[row][col]=\*ptr;

ptr++;

col++;

}

Names[row][col] = '\0';

printf("\n\n");

putchar(\*ptr);

printf("\n\n");

puts(Names[row]);

while(1){

ptr++;

row++;

col=0;

if(\*ptr == '\0')

{

flag = 1;

break;

}

while(\*ptr != '\0')

{

if(\*ptr == '|')

break;

Names[row][col]=\*ptr;

ptr++;

col++;

}

Names[row][col] = '\0';

// printf("\n\n");

// putchar(\*ptr);

printf("\n\n");

puts(Names[row]);

}

/\*

ptr++;

row++;

col=0;

while(\*ptr != '\0')

{

if(\*ptr == '|')

break;

Names[row][col]=\*ptr;

ptr++;

col++;

}

Names[row][col] = '\0';

printf("\n\n");

putchar(\*ptr);

printf("\n\n");

puts(Names[row]);

\*/

printf("\n\n");

return 0;

}

=>#include <stdio.h>

#include <string.h>

#include <stdlib.h>

/\*

struct tagName1{

mem/properities of structure

struct tagName2{

}

};

\*/

typedef struct Employee

{

int id;

int sal;

int phno;

char Name[20];

char Gender;

}EMP;

int display(EMP \*);

int main()

{

char Lines[] = "101|Amit Kumar|M|8888|10000";

char Names[10][20];

int row=0,col=0,i;

char \*ptr=NULL;

int flag = 0;

EMP e1;

EMP \*testEmp=NULL;

testEmp = (EMP \*)malloc(sizeof(EMP));

ptr = strtok(Lines,"|");

do{

strcpy(Names[row],ptr);

// puts(Names[row]);

ptr = strtok(NULL,"|");

row++;

}while(ptr != NULL);

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

printf("\n%s",Names[i]);

// names[2][0]=>'M'

// names[2][1]=>'\0'

e1.id = atoi(Names[0]);

strcpy(e1.Name,Names[1]);

e1.Gender = Names[2][0];

e1.phno = atoi(Names[3]);

e1.sal = atoi(Names[4]);

printf("\nEmployee Records are\n");

if(display(testEmp)==1)

{

printf("\nError in Display\n");

return 1;

}

printf("\n\n");

return 0;

}

/\*

rdt fName(args)

{

----

return rtd;

}

\*/

int display(EMP \*e)

{

if(e == NULL)

return 1;

printf("\nID: %d",e->id);

printf("\nName: %s",e->Name);

printf("\nGender: %c",e->Gender);

printf("\nPhNo: %d",e->phno);

printf("\nSalary: %d",e->sal);

return 0;

}

=>#include <stdio.h>

#include <string.h>

#include <stdlib.h>

typedef struct Employee

{

int id;

int sal;

int phno;

char Name[20];

char Gender;

}EMP;

int printEmp(EMP \*);

int getEmp(EMP \*);

int main()

{

EMP e1;

EMP \*e=NULL;

// e = &e1;

e = (EMP \*)malloc(sizeof(EMP));

/\*

scanf("%d%d%d",&e1.id,&e1.sal,&e1.phno);

scanf("%s",e1.Name);

getchar();

scanf("%c",&e1.Gender);

\*/

scanf("%d%d%d",&e->id,&e->sal,&e->phno);

scanf("%s",e->Name);

getchar();

scanf("%c",&e->Gender);

printf("\nID: %d",e->id);

printf("\nName: %s",e->Name);

printf("\nGender: %c",e->Gender);

printf("\nPhNo: %d",e->phno);

printf("\nSalary: %d",e->sal);

printf("\n\n");

return 0;

}

=>free():invalid pointer:

we are freeing the

=>#include <stdio.h>

#include <string.h>

#include <stdlib.h>

typedef struct Employee

{

int id;

int sal;

int phno;

char Name[20];

char Gender;

}EMP;

int printEmp(EMP \*);

int getEmp(EMP \*);

int main()

{

int noEmp;

int i;

EMP \*e=NULL;

EMP \*temp=NULL;

// e = &e1;

printf("\nEnter the Number of Employee Strength: ");

scanf("%d",&noEmp);

e = (EMP \*)malloc(noEmp\*sizeof(EMP));

temp = e;

/\*

scanf("%d%d%d",&e1.id,&e1.sal,&e1.phno);

scanf("%s",e1.Name);

getchar();

scanf("%c",&e1.Gender);

\*/

for(i=0;i<noEmp;i++,e++)

{

getEmp(e);

}

e = temp;

for(i=0;i<noEmp;i++,e++)

printEmp(e);

printf("\n\n");

e = temp;

free(e);

return 0;

}

int getEmp(EMP \*e)

{

printf("\nEnter the Employee Details\n");

scanf("%d%d%d",&e->id,&e->sal,&e->phno);

scanf("%s",e->Name);

getchar();

scanf("%c",&e->Gender);

}

int printEmp(EMP \*e)

{

printf("\nEmployee Details are\n");

printf("\n===============================\n");

printf("\nID: %d",e->id);

printf("\nName: %s",e->Name);

printf("\nGender: %c",e->Gender);

printf("\nPhNo: %d",e->phno);

printf("\nSalary: %d",e->sal);

printf("\n===============================\n");

}