**Lab 8 – Submitted by Hari Prasad B K | 2547120**

**Code**

#include<stdio.h>

#define MEDICAL\_ALLOWANCE 25000

struct Employee {

char name[50];

int age;

char designation[30];

float basic\_pay;

};

float getHRA(float basic\_pay) {

return (basic\_pay \* 25) / 100;

}

float getDA(float basic\_pay) {

return (basic\_pay \* 30) / 100;

}

float getTravelAllowance(float basic\_pay) {

if (basic\_pay >= 100000) {

return (basic\_pay \* 3) / 100;

}

return (basic\_pay \* 5) / 100;

}

float computeGrossPay(float basic\_pay) {

return basic\_pay + getHRA(basic\_pay) + getDA(basic\_pay) + getTravelAllowance(basic\_pay) + MEDICAL\_ALLOWANCE;

}

int main() {

struct Employee emps[10];

struct Employee disp;

FILE \*fptr;

printf("\nEnter the details of Employees\n-----------\n");

for (int i = 0 ; i < 10 ; i++) {

printf("\nName of employee %d: ",i + 1);

scanf("%s",emps[i].name);

printf("\nAge of employee %d: ",i + 1);

scanf("%d",&emps[i].age);

printf("\nDesignation of Employee %d: ",i + 1);

scanf("%s",emps[i].designation);

printf("\nBasic Pay: ");

scanf("%f",&emps[i].basic\_pay);

}

// Writing into file in binary

fptr = fopen("employee.dat","wb");

if (fptr == NULL) {

printf("Error opening file!");

return 0;

}

fwrite(emps,sizeof(struct Employee),10,fptr);

fclose(fptr);

// Reading from binary

fptr = fopen("employee.dat","rb");

if (fptr == NULL) {

printf("Error opening file!");

return 0;

}

for (int i = 0 ; i < 10 ; i++) {

fread(&disp,sizeof(struct Employee),1,fptr);

printf("\n-------------");

printf("\nEmployee %d:",i + 1);

printf("\nName: %s",disp.name);

printf("\nAge: %d",disp.age);

printf("\nDesignation: %s",disp.designation);

printf("\nBasic Pay: %.2f",disp.basic\_pay);

printf("\nHRA(25per): %.2f",getHRA(disp.basic\_pay));

printf("\nDA(30per): %.2f",getDA(disp.basic\_pay));

printf("\nTravelling Allowance(5per if > 1L else 3per): %.2f",getTravelAllowance(disp.basic\_pay));

printf("\nMedical Allowance(fixed): %d", MEDICAL\_ALLOWANCE);

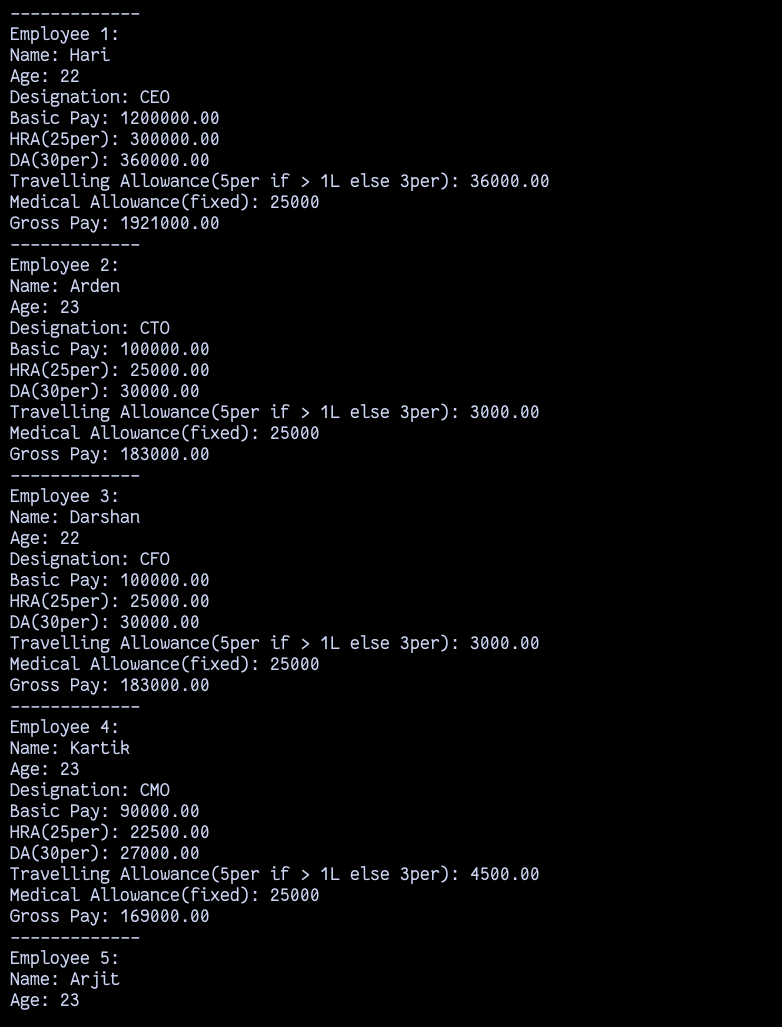
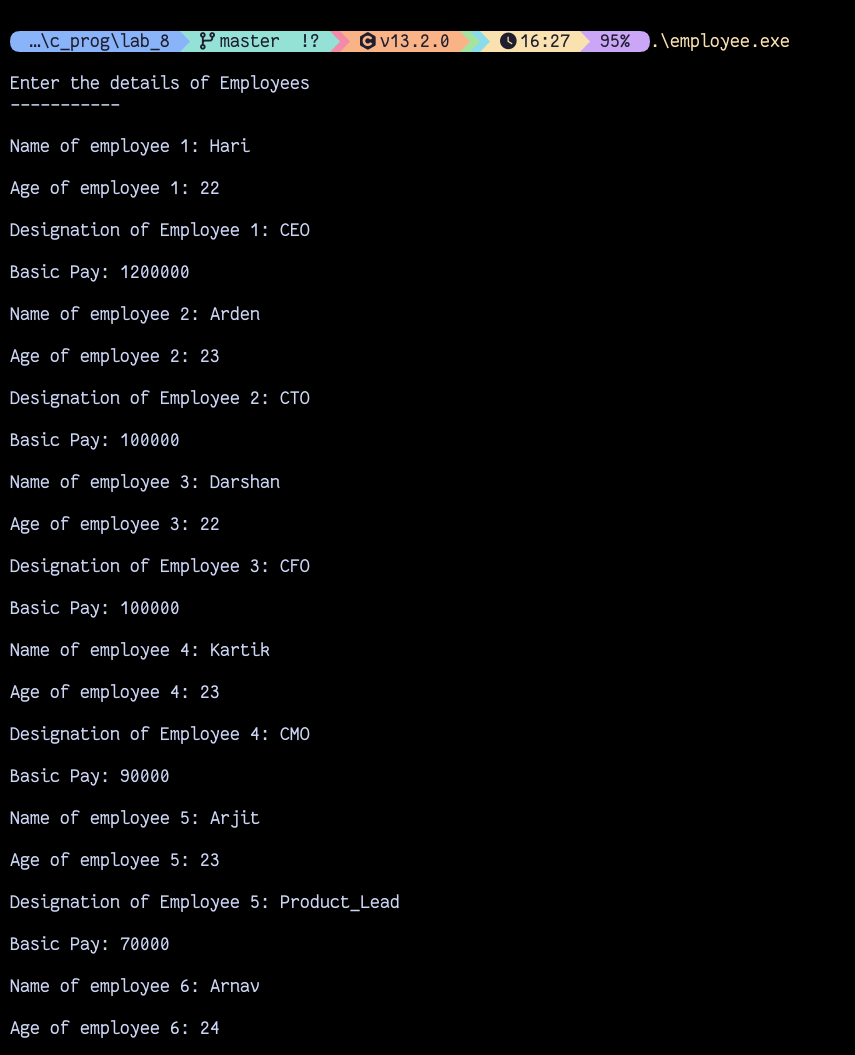
printf("\nGross Pay: %.2f", computeGrossPay(disp.basic\_pay));

}

return 0;

}

**Output**

****