**Batch: A1 Roll No.: 1911004**

**Experiment / assignment / tutorial No. 12**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

|  |
| --- |
| **TITLE: To store and print the information for Time Period and different parts of Car using structure.** |

**AIM:**

1. **Define a structure to store the following information for Time period**

* **Hours**
* **Minutes**
* **Seconds**

**Accept the information for two time period. Write a C program to add these two time periods. Also display the difference between these two time periods.**

1. **Write a C program to define a structure to store the following information of different parts of the car.**

* **Part Name**
* **Part Model**
* **Part price**

**Accept and store the information of ‘n’ parts from the user and display this information in the ascending order of the part name.**

**Expected OUTCOME of Experiment:**

**CO1. Formulate a problem statement and develop the logic (algorithm/flowchart)**

**for its solution.**

**CO2. Apply basic concepts of C programming for problem solving.**

**CO3. Illustrate the use of derived and structured data types such as arrays,**

**strings, structures and unions.**

**CO4. Demonstrate the concepts of modular programming through functions and**

**dynamic memory allocation through use of pointers.**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Books/ Journals/ Websites referred:**

1. **Programming in C, second edition, Pradeep Dey and Manas Ghosh, Oxford University Press.**
2. **Programming in ANSI C, fifth edition, E Balagurusamy, Tata McGraw Hill.**
3. **Introduction to programming and problem solving, G. Michael Schneider ,Wiley India edition.**
4. **Let’s C by Yashwant Kanetkar**
5. [**http://cse.iitkgp.ac.in/~rkumar/pds-vlab/**](http://cse.iitkgp.ac.in/~rkumar/pds-vlab/)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Problem Definition:**

1. ***Structure to store two time periods:***

**Define a structure named Time which has three integer data items i.e. hour, minute and second. In this program user is asked to enter two time periods and these two periods are stored in structure variables (eg Time1 and Time2 respectively).The task is to add the variables of the Time data type through a function void AddTime(Time \*time1, Time \*time2) which takes as arguments the addresses of two Time type variables, adds these variables and stores the result in Time2.Another function DifferenceinTime calculates the difference between the time periods and the result is displayed in main() function**

1. ***Information of parts of the car:***

**Define a structure to store the information of parts of car such as part name, model and price which are defined as members of the structure.**

**Then accept the number of parts (n) from the user and store the information about (n) parts using array of structures.**

**Then using a sorting technique, sort the given information of the parts in the ascending order i.e. alphabetical order of the part name and display the sorted information.**

**Algorithm:**

**1. //prog1**

**1. Start.**

**2. declaring and defining a structure and members like hours, minutes, seconds**

**3. input two time periods in two variable of struct Time**

**Time1.hr ; Time1.min ; Time1.sec;**

**Time2.hr ; Time2.min ; Time2.sec; // are taken as input**

**4. Function call to differenceinTime(time1,time2); to work on difference in time periods and stored them in (variable of structure Time) Dtime.**

**(For subtraction if seconds of the time period from which another time period is to be subtracted is lesser then use the borrow method of subtraction and accordingly decrement minutes similarly do the same for minutes.)**

1. **if time3.sec<time4.sec**

**time3.sec=time3.sec;**

**time3.sec=time3.sec+60;**

1. **if(time3.min<time4.min)**

**time3.hr= time3.hr -1;**

**Time3.min= Time3.min +60**

1. **DTime.hr=time3.hr-time4.hr;**

**DTime.min=time3.min-time4.min;**

**DTime.sec=time3.sec-time4.sec;**

1. **if DTime.sec>=60**

**DTime.min= DTime.min +(DTime.sec/60);**

**DTime.sec= DTime.sec %60;**

1. **if DTime.min>=60**

**DTime.hr= DTime.hr +(DTime.min/60);**

**DTime.min= DTime.min %60;**

1. **return DTime;// return entire structure DTime**
2. **if ((( DTime.min)<10)&&(( DTime.sec)<10))**

**printf(" \n\t %d : 0%d : 0%d \n", DTime.hr, DTime.min, DTime.sec);**

1. **else if (( DTime.min)<10)**

**printf(" \n\t %d : 0%d : %d \n", DTime.hr, DTime.min, DTime.sec);**

1. **else if(( DTime.sec)<10)**

**printf(" \n\t %d : %d : 0%d \n", DTime.hr, DTime.min, DTime.sec);**

1. **else**

**printf(" \n\t %d : %d : %d ", DTime.hr, DTime.min, DTime.sec);**

1. **function call to AddTimes(&Time1,&Time2); to work on sum of time periods and stored them (variable of structure Time)**

**time2**

**(After addition make sure the values of minutes and seconds are under 60 if not using logic make proper changes.)**

1. **time2->hr=time2->hr+time1->hr;**

**time2->min= time2->min+ time1->min;**

**time2->sec= time2->sec+ time1->sec;**

1. **if time2->sec>=60**

**time2->min= time2->min+ (time2->sec/60);**

**time2->sec= (time2->sec)%60;**

1. **if time2->min>=60)**

**time2->hr+=time2->min/60;**

**time2->min%=60;**

1. **if(((time2->min)<10)&&((time2->sec)<10))**

**printf(" \n\t %d : 0%d : 0%d \n",time2->hr,time2->min,time2->sec);**

1. **else if((time2->min)<10)**

**printf(" \n\t %d : 0%d : %d \n",time2->hr,time2->min,time2->sec);**

1. **else if((time2->sec)<10)**

**printf(" \n\t %d : %d : 0%d \n",time2->hr,time2->min,time2->sec);**

1. **else**

**printf (time2->hr,time2->min,time2->sec);**

1. **Display the sum and difference of time periods.**
2. **Stop.**

**2. //prog2**

1. **Start**
2. **Declare structure with members p,m,pN.**
3. **In main()**
4. **Input n ( the no of car parts);**
5. **the array of structure c[n] ,t is defined**
6. **i=0;**

**while (i<n)**

**input (c[i].pN, c[i].m, c[i].pN);**

**i=i+1;**

1. **The unsorted array is printed**

**i=0;**

**while (i<n)**

**printf (c[i].pN, c[i].m, c[i].pN);**

**i=i+1;**

1. **The sort (c) is called**

**where we use strcmp(s1,s2) to compare the strings and sort structure ascending alphabetically**

**i=0;**

**while (i<n)**

**j=0;**

**while (j<n-1)**

**if(strcmp(c1[j].pN,c1[j+1].pN)>=0)**

**t=c1[j];**

**c1[j]=c1[j+1];**

**c1[j+1]=t;**

**j++;**

**i++;**

1. **The sorted array is printed**

**i=0;**

**while (i<n)**

**printf (c[i].pN, c[i].m, c[i].pN);**

**i=i+1;**

1. **Stop**

**Implementation details:**

**1)**

#include<stdio.h>

typedef struct Time

{

int hr,min,sec;

}Time;

Time Time1,Time2,ATime,DTime;

void AddTimes(Time \*time1,Time \*time2);

Time DifferenceinTime(Time,Time);

int main()

{

printf("ENTER TIME 1 IN FORMAT HOURS : MINUTES : SECONDS\n");

scanf("%d %d %d",&Time1.hr,&Time1.min,&Time1.sec);

printf("ENTER TIME 2 IN FORMAT HOURS : MINUTES : SECONDS \n");

scanf("%d %d %d",&Time2.hr,&Time2.min,&Time2.sec);

{

printf("\n\t\t\tDIFFERENCE IN TIME\n");

DTime=DifferenceinTime(Time1,Time2);

if((( DTime.min)<10)&&(( DTime.sec)<10))

printf(" \n\t%d : 0%d : 0%d \n", DTime.hr, DTime.min, DTime.sec);

else if(( DTime.min)<10)

printf(" \n\t%d : 0%d : %d \n", DTime.hr, DTime.min, DTime.sec);

else if(( DTime.sec)<10)

printf(" \n\t%d : %d : 0%d \n", DTime.hr, DTime.min, DTime.sec);

else

printf(" \n\t%d : %d : %d ", DTime.hr, DTime.min, DTime.sec);

}

printf("\n\t\t\tADDITION OF TIME\n");

AddTimes(&Time1,&Time2);//call by reference

return 0;

}

void AddTimes(Time \*time1,Time \*time2)

{

time2->hr=time2->hr+time1->hr;

time2->min= time2->min+ time1->min;

time2->sec= time2->sec+ time1->sec;

if((time2->sec)>=60)

{ time2->min+=(time2->sec/60);

(time2->sec)%=60;

if(time2->min>=60)

{

time2->hr+=time2->min/60;

time2->min%=60;

}

}

if(((time2->min)<10)&&((time2->sec)<10))

printf(" \n\t%d : 0%d : 0%d \n",time2->hr,time2->min,time2->sec);

else if((time2->min)<10)

printf(" \n\t%d : 0%d : %d \n",time2->hr,time2->min,time2->sec);

else if((time2->sec)<10)

printf(" \n\t%d : %d : 0%d \n",time2->hr,time2->min,time2->sec);

else

printf(" \n\t%d : %d : %d ",time2->hr,time2->min,time2->sec);

}

Time DifferenceinTime(Time time3,Time time4)

{

if(time3.sec<time4.sec)

{time3.sec--;

Time3.sec+=60

} if(time3.min<time4.min)

{time3.hr--;

Time3.min+=60

}

DTime.hr=time3.hr-time4.hr;

DTime.min=time3.min-time4.min;

DTime.sec=time3.sec-time4.sec;

if((DTime.sec)>=60)

{

DTime.min+=(DTime.sec/60);

DTime.sec%=60;

if(DTime.min>=60)

{

DTime.hr+=(DTime.min/60);

DTime.min%=60;

}

}

return DTime;

}

**2)**

#include<stdio.h>

#include<string.h>

typedef struct car

{

int p;

char m[25],pN[50];//for name of part

}C;

int n,i,j; C t;

void sort(C c1[n])

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

{for(j=0;j<n-1;j++){

if(strcmp(c1[j].pN,c1[j+1].pN)>=0)

{ t=c1[j];

c1[j]=c1[j+1];

c1[j+1]=t;

} } } }

int main()

{ printf("ENTER THE NO OF PARTS \n\t\t");

scanf("%d",&n);

C c[n];

printf("ENTER THE DETAILS OF THE PARTS \n");

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

{ printf(" PART NAME : \t\n");

fflush(stdin); scanf("%s",c[i].pN); getc(stdin);

printf(" MODEL NAME : \t\n");

fflush(stdin); scanf("%s",c[i].m); getc(stdin);

printf(" PRICE : \t\n");

scanf("%d",&c[i].p); //getc(stdin);

} printf("\t\tBEFORE SORTING\n");

printf("ENTER \n PART NAME MODEL NAME PRICE \n");

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

{

printf("%s %s %d\n",c[i].pN,c[i].m,c[i].p);

}

sort(c);

printf("\t\tAFTER SORTING\n");

printf("ENTER \n PART NAME MODEL NAME PRICE \n");

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

{

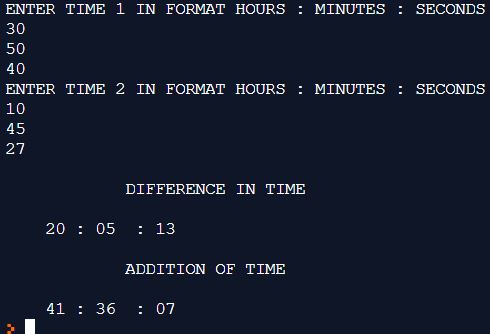
printf("%s %s %d\n",c[i].pN,c[i].m,c[i].p);

} return 0;

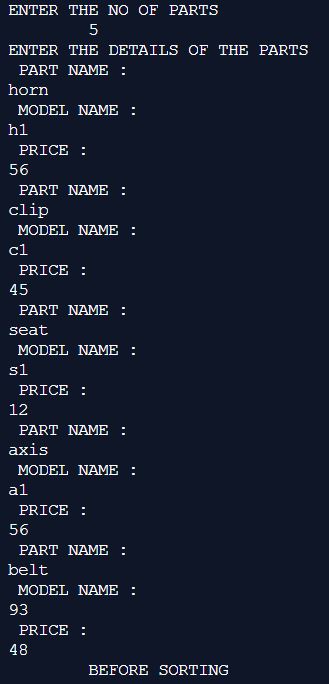
}

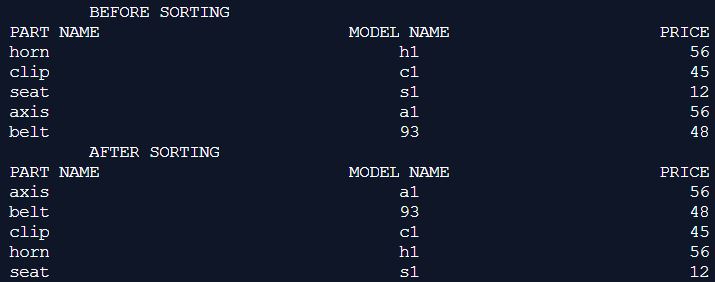
**Output(s):**

**1)**



**2)**



**(continued ss )**

**Conclusion:**

**1)The difference and addition of time period was successfully printed using structure.**

**2)The car parts were sorted in ascending alphabetical wise successfully.**

**Post Lab Descriptive Questions:**

1. **Write down the difference between the structure and union**

**ANS:**

|  |  |
| --- | --- |
| **struct (structure)** | **union** |
| **Each member of the structure is allotted different memory location.** | **Each member of the union is allotted same memory location.** |
| **Declared using**  **struct tag\_name**  **{**  **//datatype members;**  **}variables;** | **Declared using**  **Union union\_name**  **{**  **//datatype members;**  **}variable;** |
| **The size of structure= total size of the members of the structure** | **The total size of the union = the size of the largest member of the structure;** |
| **It store multiple values of the different members of the structure.** | **It store single value at a time of all the members of the union** |

1. **Explain the meaning and purpose of following:**
2. **typedef keyword**
3. **sizeof operator**
4. **Bit fields**

**ANS:**

1. **typedef keyword**

**It helps to define and give the symbolic names to primitive and user defined datatype .Thus this name becomes the keyword referring to that datatype. It thus creates the synonym (alternate names )of that datatype. It reduces the complexity in the program.**

**Eg.**

**typedef unsigned char byte;**

**int main()**

**{ byte b1, b2;**

**b1 = 'c';**

**printf("%c ", b1);**

**return 0;**

**}**

**//this will create 8bit data type byte**

1. **sizeof operator**

**This operator helps us to calculate the size of operand(array,structure,variables) in memory. It is a compile time unary operator .it gives us unsigned integral value of byte.Thus works for both primitive ,user defined datatypes.**

1. **Bit fields**

**It consists of a number of adjacent computer memory locations which have been allocated to hold a sequence of bits, stored so that any single bit or group of bits within the set can be addressed. A bit field is most commonly used to represent integral types of known, fixed bit-width.**

1. **Write the output of the given code with justification**

**#include<stdio.h>**

**int main()**

**{ struct org**

**{ int employees;**

**char comp[5];**

**struct founder**

**{**

**char ceo[10];**

**}p;**

**};**

**struct org x = {4000, "Ford", "Henry"};**

**printf("%s %d %s", x.comp, x.employees, x.p.ceo);**

**return 0;**

**}**

**ANS:**

**Ford 4000 Henry**

**Here we are using structures. The struct org has three members int emplyees, char comp[5], struct founder p.**

**p is variable of the founder struct with member char ceo[10].**

**x is the variable of struct org**

**struct org x = {4000, "Ford", "Henry"};**

**here we initialize the**

**x.employees=4000;**

**strcpy(x.comp,”Ford”);**

**strcpy(x.p.ceo,”Henry”); then we print it.**

**Date: 03-05-2020 Signature of faculty in-charge**