OS ASSESMENT 2

#include<stdio.h>  
#include<conio.h>  
#include<process.h>  
#include<string.h>

//The #includes are used to include header files e.g (file\_name.h) which have built in functions in them that can be used throughout the file. e.g #include <stdio.h> allows the user to use functions such as printf() and scanf() in their program.

//start of the main function

void main()  
{

//declare a 2d character array of name p with 10 rows and 5 columns.   
                char p[10][5];

//declare a 1d array of name et, wt and count.pt all of size 10 (i.e 10 rows)  
                int et[10],wt[10], count,pt[10];

//declare integers “timer” and assign value of 3, “rt”, “I”, “j”, “totwt” and assign value of 0 to all, “n” and assign value of 5, “found” and assign value of 0 and finally m.

Int timer=3 ,rt,i,j,totwt=0,t,n=5,found=0,m;

//declare float avgwt  
                float avgwt;

//call the clear screen function  
                clrscr();

//start a loop that uses a variable I set to 0, as an index that increments each time an iteration occurs until I reaches the value-1 in N (4) and then it breaks out of the loop

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

//Display a message to a user telling them to enter the name of the process and also how long that process will run for and then read in the values from the keyboard and store them in the 2d array “p” where I is the index of the element stored.

                                printf("enter the process name : ");  
                                scanf("%s",&p[i]);  
                                printf("enter the processing time : ");  
                                scanf("%d",&pt[i]);  
                }

//set m to the value that n now holds, initialise the first element of the array “wt” to contain the number 0 and also set I back to 0 to be used in the next loop.

                m=n;  
                wt[0]=0;  
                i=0;

//start a do while loop where the loop iterates each time, provided that the condition (i<n) or (i< 5) has not been met.

                do  
                {

//check in each iteration of the loop if the nth element of the array pt (e.g if I = , check element pt[1]) to see if it is greater than the value in timer (3) and if it is, run the code in the if statement, otherwise, run the code in the else statement

                                if(pt[i]>timer)  
                                {

//set the variable “rt” equal to the value that is stored in the nth element of “pt” (pt[i]) and take the value that is in timer away from it. Then copy the string from the array p with index of I into the array p with index of n (e.g contents of p[2] are now copied into p[n]). Then replace the contents of the array pt with index I with variable “rt”. Then replace the contents of the array et with index I with variable “timer”. Then finally increment n

                                                rt=pt[i]-timer;  
                                                strcpy(p[n],p[i]);  
                                                pt[n]=rt;  
                                                et[i]=timer;  
                                                n++;         
                                }  
                                else

//set array et[i] equal to contents of pt[i]

                                                et[i]=pt[i];

//increment I for next iteration, set contents of wt[i] equal to the contents of the previous element in wt, plus the contents of et at index I-1 and finally check the condition to see if loop can run again.

                                i++;  
                                wt[i]=wt[i-1]+et[i-1];  
                }while(i<n);

//set variable count equal to 0 and start another loop using index of I set to 0, condition of I less than m and so that I increments on each iteration of loop. Within that loop, create another loop using an index of J set to the value of I in the first loop (will change on each iteration of I), condition of n less than or equal to n and so that j is incremented on each iteration.

                count=0;  
                for(i=0;i<m;i++)  
                {  
                                for(j=i+1;j<=n;j++)  
                                {

//using the strcmp function to compare two strings (those stored in p[i] and p[j]), if the two strings match each other, increment the count value and set the variable “found” equal to the contents of j.

                                                if(strcmp(p[i],p[j])==0)  
                                                {  
                                                                count++;  
                                                                found=j;  
                                                }  
                                }

//checks if the contents of “found” is not 0 and if this is true it sets the contents of wt[i] equal to the contents of wt at position found take away the variable count multiplied by the variable timer, then set count and found back to 0 for next iteration.

                                if(found!=0)  
                                {  
                                                wt[i]=wt[found]-(count\*timer);  
                                                count=0;  
                                                found=0;  
                                }  
                }

//create a new loop which sets the value of “totwt” equal to itself plus the value in wt[i]

                for(i=0;i<m;i++)  
                                totwt+=wt[i];

//set the contents of “avgwt” equal to “totwt” divided by “m” and cast it to a float so that decimal points can be stored if the answer has a remainder

                avgwt=(float)totwt/m;

//create a loop to print out the variables stored in each process (p[i],pt[i] and wt[i]) for each instance of I up to I < m

                for(i=0;i<m;i++)  
                                printf("\n%s\t%d\t%d",p[i],pt[i],wt[i]);

//print messages to tell the user the total waiting time (totwt) and the total average time (avgwt)  
                printf("\ntotal waiting time %d\n",totwt);  
                printf("total avgtime %f",avgwt);

//get an input from the keyboard to keep the file open

        getch();  
}