Introduction to Basic Language :

BASIC language is a high level language invented by KEMENY and KURTZ in 1967. The other high level languages used in science are FORTRAN, Pascal, C and C++, Java C# (called C sharp). One of the important applications of high level languages like BASIC, FORTRAN and C is to compute values of parameters expressed by scientific formulas. A programming language like BASIC uses a set of alphabets (upper case letters) and special symbols like +, -*, /" ; ^ % & and so on.

Let us consider the BASIC language and analyse its features. Each BASIC statement (when executed by an interpreter) has a statement number keyword and instruction. The program contains variables and constants. Variables are of two types (**numeric** and **string**) Similarly the constants are of two types (**string** and **numeric**). In mathematics we are familiar with the equation y = ax where y and x are variables a is a constant. In BASIC language we have numeric and string variables. A computer program has the following sections input statements assignment statements, expression evaluation and print statements.

High Level Languages

High level languages are developed to allow application programs, which are machine independent. High level language permits the user understandable codes using the language structure. In order to execute a high-level language program, it should be translated into a machine language either using a compiler or interpreter.

Let us consider the simple program to convert temperature in Celcius to Kelvin temperature once again in more detail.

```
10  REM CONVERSION OF CELSIUS TEMP TO KELVIN TEMP
20  INPUT "GIVE NAME OF PLACE AND TEMP IN CELSIUS" ; N$, CEL
30  ABZ = 273
40  KEL = CEL + ABZ
50  PRINT "THE TEMPERATURE OF " ; N$ ; "IS =" ; "KEL ; " ; "KELVIN "
60  END
```

In this program KEL, ABZ, CEL are numeric variables. N$ is a string variable. 273 is a numeric constant. "THE TEMPERATURE OF", "IS =", "KELVIN ", "GIVE NAME.OF PLACE AND TEMP.IN CELSIUS" are string constants. When this program is executed with N$=PHYSICAL CHEMISTRY LABORATORY and CEL = 25 The output result is THE TEMPERATURE OF PHYSICAL CHEMISTRY LABORATORY IS 298 KELVIN.

C languages

The C language invented by Dennis Ritchie in the 1970's has many interesting features which are presented below :

* Question mark operator
* concise for - loop
* Recursion
* Break
* Continue
* Pointers
* Structures
* Unions
* Macros

To execute all C programs using TURBO C compiler the following files have to be included.

```
#include<stdio.h>
#include<math.h>
```

The structure of a C program is shown below :

One of the very important features of C language is the question mark operator which is equivalent to the IF-THEN ELSE statement. The form of the statement with this operator is Variable = (condition) ? exp1: exp 2;

If condition is true variable is assigned to expression 1 and if conditjon is false variable is assigned Exp2. The following program calculates reduced mass of a homo diatomic or hetero diatomic molecule. The formulas for reduced masses of homo and hetero diatomics are given below :

$\mu(homo) M_1/2N_0$ $\mu(hetero) = M_1 * M_2/((M_1+M_2)*N_0)$

Question Mark Operator

Program 1

```
/* Program to calculate reduced mass of homo diatomic or hetero
diatomic. The number of */
/* inversion centers in the molecule is input. If number of inversion
centers is Zero */
/* the calculation is for hetero diatomic and if number of inversion centers
is one */
/*the calculation is for homo diatomic*/
#define AVO 6.023e23
main()
{
    char molecule[20];
    int nin;
    double redm, m1, m2;
    printf("enter name of molecule/n");
    gets (molecule);
    printf("enter no. of inv. centers and masses\n");
    scanf("%d%if", &nin, &m1, & m2);
    redm=(nin==0)?m1*m2/((AVO*(m1+m2):m1(2.0*AVO);
    printf("the reduced mass for %s is % 10.4e grams\n", molecule, redm);
}
```

Input and output

```
enter name of molecule
hydrogen
enter no. of inv. centers and masses
1.1.008 35.45
the reduced mass for hydrogen is 8.3679e-25 grams
enter name of molecule
hydrogen_chloride
enter no. of inv. centers and masses
0 1.008 35.45,
the reduced mass for hydrogen_chloride is 1.6273e-24 grams
```

The advantage of C is that the 4 statements in FORTRAN or BASIC

```
30 IF NIN=0 THEN 40 ELSE 70
40 REDM=M1*M2/((AVO*(M1+M2))
50 GO TO 80
60 REDM = M1/(2*AVO)
80....
```

be replaced by the single statement
redm=(nin==0)?m1*m2/((AVO*(m1+m2):m1/(2.0*AVO);
in C language.

Concise for loop

Another example in chemistry is to compute the work of reversible isothermal or adiabatic expansion of ideal monoatomic gas using the question mark operator The program listing and output are presented given below.

The next important feature of C language is the concise for loop.

Program 2

This program calculates the molecular weight of a tetra atomic system ABCD (KCNS) from data on atomic weights of the 4 atoms.

```
double awts[4]={39.0, 12.01, 14.008, 32.0};
main()
{
    char molecule[20];
    int i;
    printf("enter name of molecule\n");
    gets(molecule);
    for.(i=0;mw=0.0;i<3;mw+=awt[i],++i);
    printf("the mol.wt.of %s is %7.3f grams/n", molecule, mw);
}
```

Input and Output

```
Enter name of molecule
KCNS
The mol.wt of KCNS is 97.018 grams
```

Unlike FORTRAN and BASIC the for loop in C can have multiple loop variables. The same program when written in BASIC requires 4 statements for the for loop as shown below:

```
30 MW=0.0
40 FORI=1 TO 4
50 MW=MW+AWT(I)
60 NEXTI
70....
```

Global and local variables

The program introduces the global and local variables and illustrates the use of global variable. Similar to subroutines we have function subprograms in C. In such programs only one output variable is returned back to the main program. Let us see the following function subprogram which calculates the temperature in Celsius from Kelvin temperature and absolute zero value.

Program 3

```
/*main program*/
main()
{
      int kelvin,ab_ze=273.celsius,Z;
      printf("enter Kelvin temp.\n");
      scanf("%d", & Celsius);
      z=sub(kelvin, ab_ze);
      printf("the Celsius temperature is %d degrees\n",z);

}
/*Function program*/
sub(x,y)
int x,y;
{
      return(x-y);
}
```

The variable ab_ze is copied onto argument y and the variable kelvin is copied on to x and the function program sub(x,y) evaluates x-y and returns it to the main program. A return statement returns only one output value to the main. If we have a problem involving simultaneous equations in two unknowns this function program cannot be used and we can make use of global variables. The following program to calculate the Arrhenius parameters Ea and A illustrates this point.

Program 4

The Arrhenius equation for temperature dependence of rate constant is

$k = \exp(-E_a/RT)$

For two temperatures this equation is written as

$\ln k_1 = \ln A - E_a/RT_1$
$\ln k_2 = \ln A - E_a/RT_2$

These two simultaneous equations in the unknowns Ea and lnA can be solved and the program is written using global variables.

```
/*main program*/
#define R=8.314
double xx,yy,k1, k2, t1, t2;/*global variables*/
main()
{
    char reaction[30];
    double eact.pref;
    printf("give the name of the reaction\n");
    gets(reaction);
    printf("give k1, k2, t1 and t2\n");
    scanf("%if, &k1, &k2, &t1, &t2);
    sim_eq();
    pref=exp(xx);
    eact=yy;
    printf("the Arrhenius Parameters for %s are:\n", reaction);
    printf("_____\n");
    printf("the energy of activation is % 10.4e joules \n", eact);
    printf("the preexponential factor is %10.4e per sec\n", pref);
    printf("_____\n");
}
/*function program*/
/*The simultaneous equations ax+by=c; and dx+ey=f have solutions*/
/*x=ce-bf/(ae-bd); y=af-cd/(ae-bd);*/
sim_eq()/*function definition*/
{
    double a1, b1, c1, d1, e1, f1, z1;
    a1=1.0;
    b1=-1.0/(R*t1);
    c1=logk1;
    d1=1.0;
    e1=-1.0/(R*t2);
    f1=logk2;
    z1=a1*e1-e1-b1*d1;
    xx=(c1*e1-b1*f1)/z1;
    yy=(a1*f1-c1*d1)/z1;
}
```

Input and Output

```
Give the name of the reaction
Decomposition_of_benzaldehyde
Give k1, k2, t1 and t2
0.011 145.0 700.0 1000.0
the Arrhenius Parameters for Decomposition_of_benzaldehydeare
the energy of ectivation is 1.8403e5 joules
the preexponential factor is 5.9518e11 persec
```

In this program xx and yy are global variables which can be used in main program as well as function program. Thus two output variables are taken to the main program using the technique of global variables. Another way to transfer more than one output variable to the main is by using pointers. The next program illustrates the use of pointers to obtain Arrhenius parameters.

Pointers and their use

Program 5

```
#define R=8.314
main()
{
    char reaction [30];
    double eact,pref, a,b,c,d,e,f,*a1, *b1, *c1, *d1, *e1, *f1,
    /* *a1, *b1, *c1, *d1, *e1, and *f1 are pointers to the variables
a,b,c,d,e and f*/
    double 1, k2, t1, t2, x, y, & *x1, *y1, eact, pref;
    a1 = *a ; b1 = *b ; c1 =*c; d1 = *d ; e1 = *e; f1 = *f ; x1 = *x ; y1 = *y ;
    printf("give the name of the reaction/n");
    gets (reaction);
    printf("give k1, k2, t1 and t2/n");
    scanf ("%if%if%if%if", &k1, &k2, &t1, &t2);
    a = 1.0;
    b = -1.0(R*t1);
    c = log k1;
    d = 1.0;
    e = -1.0/(R*t2)
    f = logk2;
    sim_eq(a1, b1, c1, d1, e1, f1, x1, y1);
```

```
    pref = exp(xx);
    eact = yy ;
    printf("the ArrheniusParameters for %s are :/n", reaction);
    printf("                                    /n");
    printf("the energy of activation is % 10.4e joules/n", eact);
    printf("the preexponential factor is % 10.4e per sec/n",pref);
    Printf("\n\n");

}
/*function program*/
sim_eq(aa1, bb1, cc1, dd1, ee1, ff1, xx1, yy1) / * function definition*/
double *aa1, *bb1, *cc1, *dd1, *ee1, *ff1, *xx1, *yy1 ;

{
    double z1;
    a1 = 1.0;
    b1 = -1.0/(R*t1);
    c1 = log k1;
    d1 = 1.0;
    e1 = - 1.0 / (R*t2);
    f1 = logk2;
    z1 = (*aa1)*(*ee1) - (*bb1)*(*dd1) ;
    *xx1 = ((*cc1)*(*ee1) - (*bb1)*(*ff1)/z1;
    *yy1 = ((*aa1)*(*ff1) - (*cc1)*(*dd1)/z1 ;
}
```

Input and output

```
Give the name of the reaction
Decomposition_of_benzaldehyde
Give k1, k2, t1 and t2
0.011 145.0 700.0 1000.0
the ArrheniusParameters for Decomposition_of_benzaldehyde are
the energy activation is 1.8403e5 joules
the preexponential factor is 5.9518e11 persec
```

Two more important features of C language are the break and continue statements. The break statement is used to quit from the loop based on a condition and execute statements following the loop. The break is used to quit from innermost loop only in nested loops. The continue statement is used to skip a particular iteration in a loop. The program computes NMR $\nu = \mu H/Ih$

Break Statement

Program 6

```
/*Use of break statement in  loops*/
/* It is possible to quit from the loop and transfer the control */
/* to the statement following the loop */
/* using a condition.  In this program the NMR frequencies of nuclei with *
/* spin = 1/2 only are calculated. If the nucleus has quadrupole moment then*/
/* its NMR frequency is not computed since such nuclei have spin */
/* greater then 1/2 */
# define H 6.626e-34
# define BN 5.05e-27
# define CF 1.0e06
main ()
{
      Char nucleus [20];
      int i ;
      double field = 1.0, moment, spin, qmom, fre;
      for (i=1 ; i<=5; ++i){
      printf ("enter name of nucleus\n");
      gets(nucleus);
      printf("enter value of quadrupole moment\n");
      scanf("%if", &qmom);
      if(qmom!=0.0){
      printf("the nucleus has quadrupole moment and control exits the  loop\n");
      break ; }
      printf("enter mag.moment and spin values\n");
      scanf("%if%if", &moment, & spin);
      fre = fabs(moment)*field/(spin*H*CF);
      printf("the NMR frequency for %s is %7.3f MHz\n", nucleus,fre);
      }
}
```

Input and Output

```
Enter name of nucleus
Hydrogen
Enter value of quadrupole moment
0.0
enter magnetic moment and spin values
2.79285 0.5
```

e NMR frequency for hydrogen is 42.571MHz
ter name of nucleus
allium_69
ter value of quadrupole moment
2318
e nucleus has quadrupole moment and control exits the loop

ontinue Statement

The continue statement is used is the loops, to skip a part of the op and continue the next cycle of the loop. While the break statement bandons the loop and transfers the control out of the loop the continue tatement from stars the control to the next cycle of the loop.

rogram 7

```
The program uses continue statement*/
When this statement is used the specific iteration is skipped if condition
 true and */
the control is transferred to carry out the next iteration of loop*/
This program counts the total number of fermions between atomic
umbers 1 and 6*/
A fermion has odd number of nucleons and a Boson has even number
 nucleons*/
ain()

   char atom[20];
   int atnr, nenr, massnumber, count=0;
   for(atnr=1;atnr<=6;++atnr){
   printf("enter name of atom\n");
   gets(atom);
   printf("enter no. of neutrons\n");
   scanf("%d", & nenr);
   massnumber=atnr+nenr;
   if(massnumber%2==0){
   printf("the number of nucleons is even and");
   printf("nucleus is a Boson.control skips iteration\n");
   continue;
   }
   ++count;
   printf("the total number of nucleons between atomic numbers 1
d 6 is %d\n", count);
```

Input and Output

```
Enter name of atom
Hydrogen
Enter no.of neutrons
0
enter name of atom
Helium
Enter no. of neutrons
2
the number of nucleons is even and nucleus is a Boson. Control skips
this iteration
enter name of atom
lithium
enter no. of neutrons
4
enter name of atom
beryllium
enter no. of neutrons
5
enter name of atom
boron
enter no. of neutrons
6
enter name of atom
carbon
enter no. of neutrons
6
the number of nucleons is even and nucleus is a Boson. Control skips
this iteration

the total number of fermions between atomic numbers 1 and 6 is 4
```

A very important feature of C, C++, Pascal, FORTRAN 90 and Java is recursion. In recursion a function can call itself. Recursion is applicable to calculations when something can be defined in terms of itself. Examples are summation of numbers of finding factorials of numbers. The factorial of a number is written as

!n=n(!(n-1)). For example!4=4.!3. The following program computes the number of valence diagrams for a conjugated cyclic system.

Enter no. of pi electrons
10
enter name of system
naphthalene
the number of valence diagrams for naphthalene is 42
Enter no. of pi electrons
14
enter name of system
anthracene
the number of valence diagrams for anthracene is 429

Examples of simple chemistry programs :

The relevant formulas for these applications are given below.

1 a) and b) Celsius to Kelvin temperature:-BASIC version and C version.

Kel=CEL+273

2. Beer Lambert Law

OD= εc1

3. Molecular weights from atomic weights

$M_{KCNS} = AWT_K + AWT_C + AWT_N + AWT_S$

4. $W_{ISO} = RTln(V_2 / V_1)$

$W_{adi} = C_v(T_1 - T_2)$

5. res = res = n!((n/2)!*((n/2+1)!)

6. In the least squares method Q the linear least squares parameter is defiend as

$Q = \Sigma wi\ (yi - a0 - a1xi)^2$

With a0 and a1 given by

$a0 = [\Sigma wixi^2\ \Sigma wiyi - \Sigma wixi\ \Sigma wixiyi]/D$

$a1 = [\Sigma wi \Sigma wixiyi - \Sigma wixi \Sigma wiyi]/D$

D is given by

$D = \Sigma wi \Sigma wixi^2 - (\Sigma wixi)^2$

The standard deviation σ is

$\sigma = (Q[n-2])^{½}$

Standard deviations of a0 and a1 are given by

$da0 = (\sigma^2\ \Sigma wi/D)^{½}$

$da1 = (\sigma^2 \Sigma wixi^2/D)^{½}$

7. Molecular weight of organic compound containing C,H,S,Cl,N and O atoms MWT = N1AWC+N2AWO+N3AWHH+N4AWCI+N5AWS+N6AWN

Where N1 to N6 refer to number of atoms of the elements C,H,S,Cl,N and O. AW1 to AW6 refer to the atomic weights of these elements.

The listings of 8 Programs (1 BASIC Program and 7C programs) along with output data are presented in the next few sheets.

Summary

An introduction to computers, the block diagram of a computer, algorithms and flowcharts, BASIC Programming, Principles of C language and exmaples of programs executed using TURBOC compiler are presented.

The following are some assignment problems

Assignments on computer Programming

1. Write a BASIC Program to calculate the temperature in Fahrenheit from data of temperature in Celsius. Use formula F=(9/5)C +32 Calculate the temperature for Toulouse hall with AC(temp. 15 degrees Celsius)
2. Write a BASIC Program to calculate zero point energy form wavenumber data. Use ZPE=0.5hcwnN_0. Calculate forHCl with wn=3000 cm^{-1}.
3. Write a C Program using SWITCH statement to calculate $t_{1/2}$ for ZERO ORDER, FIRSTORDER and SECOND ORDER REACTIONS.

Use the following data:

Name of reaction	formula	initial conc.	Rate cons
Zero order	a/2K	0.01	0.005
First order	0.693/K	0.01	0.0004
Sec. order		0.01	0.005

4. write a C program to calculate the reduced mass for 4 homodiatomics using for loop.

Use the following data:

Name of molecule	atomic weight of atom
Nitrogen	14.008
Fluorine	19.0
Hydrogen	1.008
Chlorine	35.54

Formula is REDM=awt/$2N_0$. Use FOR NEXT Statement

5. Use the concise for loop to calculate molecular weight of HCN
6. Use recursion to find the molecular weight of KCNS from atomic weight data

```
/* pgm to temperature conversion*/
#include <stdio.h>
#include <math.h>
#include <conio.h>
#define ABZ 273.0
main()
{
      char place[20];
      float cel,kel;
      printf("\n\tEnter Name of Place: ");
      gets(place);
      printf("\n\n\tEnter CELSIUS Temperature: ");
      scanf("%f",&cel);
      kel=cel+ABZ;
      printf("\n\n\tThe Temperature for %s is %6.3f Degrees\n",place,kel);
      getch();
      return(0);
}
```

Input :

Enter Name of Place : Chemistry Lab
Enter CELSIUS Temperature: 27^0C

Output :

Temperature = 300 K

```
/* Pgm for BEER LAMBERT Law using C*/
#include <stdio.h>
#include <math.h>
#include <conio.h>
void main()
{
     char comp[20];
```

```
    double od,mab,len,con;
    clrscr();
    printf("\nEnter name of COMPLEX : ");
    gets(comp);
    printf("\nEnter values of od, mab and len :");
    scanf("%lf%lf%lf",&od,&mab, &len);
    con = od / (mab * len);
    printf("\nThe concentration of %s is =%10.4e Molar",comp,con);
    getch();
}
```

Input:
Enter name of COMPLEX : $K_4[Fe(CN)_6]$
Enter values of od, mab and len : 1, 7 & 2
Output:
The concentration of (complex) is = 4 Molar

```
/* MolecularWeights from Atomic Weights */
/* M(KCNS) = AWT(k) + AWT(c) + AWT(n) + AWT(s) */
#include <stdio.h>
#include <conio.h>
double awt[5] = { 0.0,39.0, 12.01, 14.008, 32.0};
main()
{
    char molecule[20];
    int i;
    double molwt;
    clrscr();
    printf("\nEnter Name of Molecule:");
    gets(molecule);
    for (i=1,molwt=0.0; i<5; molwt+=awt[i], i++);
    printf("\nThe Molecular Weight of %s is = %6.3f Grams", molecule, molwt);
    getch();
    return(0);
}
```

Input :
Moleculer Weights : 0.0,39.0, 12.0, 14.008, 32.0
Output : 97.018 gm

```
    double od,mab,len,con;
    clrscr();
    printf("\nEnter name of COMPLEX : ");
    gets(comp);
    printf("\nEnter values of od, mab and len :");
    scanf("%lf%lf%lf",&od,&mab, &len);
    con = od / (mab * len);
    printf("\nThe concentration of %s is =%10.4e Molar",comp,con);
    getch();
}
```

Input:
Enter name of COMPLEX : $K_4[Fe(CN)_6]$
Enter values of od, mab and len : 1, 7 & 2
Output:
The concentration of (complex) is = 4 Molar

```
/* MolecularWeights from Atomic Weights */
/* M(KCNS) = AWT(k) + AWT(c) + AWT(n) + AWT(s) */
#include <stdio.h>
#include <conio.h>
double awt[5] = { 0.0,39.0, 12.01, 14.008, 32.0};
main()
{
    char molecule[20];
    int i;
    double molwt;
    clrscr();
    printf("\nEnter Name of Molecule:");
    gets(molecule);
    for (i=1,molwt=0.0; i<5; molwt+=awt[i], i++);
    printf("\nThe Molecular Weight of %s is = %6.3f Grams", molecule, molwt);
    getch();
    return(0);
}
```

Input :
Moleculer Weights : 0.0,39.0, 12.0, 14.008, 32.0
Output : 97.018 gm

```
/* Pgm to Demo Question Mark Operator In C */
/* Reversible expansion of a monoatomic GAS */
#include <stdio.h>
#include <math.h>
#include <conio.h>
#define R 8.314
void main()
{
      char gas[20],type[20];
      double t1,t2,cv,v1,v2,w;
      clrscr();
      printf("\n Enter Name of a Gas:");
      gets(gas);
      printf("\n Enter Type of Expansion:");
      gets (type);
      printf ("\n Enter values of t1,t2,v1,v2:");
      scanf("%lf%lf%lf%lf",&t1,&t2,&v1,&v2);
      cv = 1.5 * R;
      w = (t1==t2) ? R*t1*log(v2/v1) : cv*(t1-t2);
      printf("\n The Work of %s for %s is=10.4e Joules",type,gas,w);
      getch();
}
```

Input:

Enter Name of a Gas : Oxygen
Enter Type of Expansion : Isothermal
Enter values of t1,t2,v1,v2 : 273, 373, 2, 4

Output:

The Work of Isothermal expansion for oxygen is= 5.7632 Joules

```
/* RECURSION */
#include <stdio.h>
#include <math.h>
#include <conio.h>
main()
{
      char system[20];
      int res,m,p,n;
      float fact();
```

```
    clrscr();
    printf("\n\n\tGive Name of System: ");
    gets(system);
    printf("\n\tEnter number of Pi Electrons:");
    scanf("%d",&n);
    m=n/2;
    p=m+1;
    res = fact(n) / (fact(m) * fact(p));
    printf("\n\tThe No of Res Structures for %s is Equal to %d",system,res);
    getch();
    return(0);
}

float fact(i)
int i;
{
    if (i==0)
    return(1);
     else
    return(i*fact(i-1));
    }
```

Input :

Name of the System : Phenthrane

No. of Pi Electrons : 10

Output :

No. of resonance structures : 429

```
/* Linear Least Squares - fit method */
#include <stdio.h>
#include <math.h>
#include <conio.h>
#define R 8.314
double a[10],b[10];
main()
{
     char system[40];
     int n,n1,kk,i,n2;
```

```
    double t[10],k[10],c[10],sum1,sum2,sum3,sum4,sum5,sum6,d,std,
std1,std2;
double z2,z3,z4,z5,z6,z7,z8,z9,s1,s2,s3,s4,intercept,slope,pre_fac,eract;
    printf("enter no of systems for computation\n");
    scanf("%d",&n1);
    printf("linear least squares fit analysis for");
    printf("variation with temperature\n");
    for(i=1;i<=n1;++i)
    printf("entername of reaction and no of data\n");
    for(i=1;i<=n1;++i)
    {
        printf("entername of reaction and no of data\n");
        scanf("%lf",&t[i]);
        a[i]=1.0/t[i];
        printf("enter value of rate constant:");
        scanf("%lf",&t[i]);
        b[i]=log(k[i]);
    }
    clrscr();
    if(n1<=0)
    {
        for(i=1;i<=kk;++i)
        c[i]=1.0;
    }
    else
    {
        for(i=1;i<=kk;++i)
        c[i]=1.0/b[i];
    }
    sum1=0.0,sum2=0.0;sum3=0.0;
    sum4=0.0,sum5=0.0;
    for(i=1;i<=kk;++i)
    {
        z2=c[i]*a[i]*a[i];
        z3=c[i]*b[i];
        z4=c[i]*a[i]*b[i];
        z5=c[i]*a[i];
        sum1+=c[i];
```

```
        sum2+=z2;
        sum3+=z3;
        sum4+=z4;
        sum5+=z5;
        z6=sum2*sum3-sum5*sum4;
        z7=sum1*sum2-sum5*sum5;
    }
    intercept=z6/z7;
    z8=sum1*sum4-sum5*sum3;
    slope=z8/z7;
    sum6=0.0;
    for(i=1;i<=kk;++i)
    {
        z8=b[i]-intercept-slope*a[i];
        z9=z8*z8*c[i];
    }
    sum6+=z9;
    d=(double)kk;
    s1=sum6/(d-2.0);
    ovd=sqrt(s1);
    s3=s1*sum2/z7;
    std1=sqrt(s3);
    s4=s1*sum1/z7;
    std2=sqrt(s4);
    pre_fac=exp(intercept);
    enact=-slope*R;
    printf("The preexponential factor and");
    printf("energy of activation");
    printf("for%s",system);
    printf("are%10.4eM-1s-1and%10.4ejoules\n",pre_fac-enact);
    printf("standard devision of slope is %10,4e\n",std1);
    printf("standard deviation of intercept");
    printf("is%10.4e\n",std2);
    printf("over all standards devision is %10.4e/n",ovd);
    getch();
```

```
/*Molecular weights of organic compounds having C,H,N,S,O and */
/*halogens*/
#include<stdio.h>
#include<math.h>
#include<conio.h>
main()
{
        char molecule[20];
        int. i,n[6];
        float awt[6],molwt;
        printf("enter name of molecule\n");
        molwt=0.0;
        for(i=1;i<=6;++i)
        {
        printf("enter at.wt of ith element\n");
        scanf("%f",&awt[i]);
        printf("enter number of atoms of ith element\n");
        scanf("%d",&n[i]);
        molwt+=n[i]*awt[i];
        }
        clrscr();
        printf("the mol.wt.of %s is", molecule);
        printf("=%7.3f grams\n",molwt);
        getch();
        return(0);
}
```