|  |  |
| --- | --- |
| Sr.No |  |
| 2.1 | Write a program to calculate the Net Salary  Allowances: DA = 70% of Basic Salary, HRA = 7% of Basic Salary, MA = 2% of Basic Salary, TA = 4% of Basic Salary, Salary Deduction : PF = 12% of Basic Salary, IT = any value (e.g. 500)  Net Salary = Basic Salary + Allowances - Deduction |
| Code (Input) | #include<stdio.h>  #include<conio.h>  void main()  {  float BasicSalary, Allowances, DA, HRA, MA, TA, Deduction, PF, IT, NetSalary;  clrscr();  printf(“Enter Basic Salary: ”);  scanf(“%f”, &BasicSalary);  printf(“Allowances:\n”);  DA=BasicSalary\*0.7;  Printf(“DA = %f”,DA);  HRA=BasicSalary\*0.07;  printf(“\nHRA = %f”,HRA);  MA=BasicSalary\*0.02;  printf(“\nMA = %f”,MA);  TA=BasicSalary\*0.04;  printf(“\nTA = %f”,TA);  printf(“\nDeduction:\n”);  PF=BasicSalary\*0.12;  printf(“Pf = %f”,PF);  printf(“\nIT = ”);  scanf(“%f”,&IT);  Allowances=DA+HRA+MA+TA;  Deduction=PF+IT;  NetSalary=BasicSalary+Allowances-Deduction;  printf(“Net Salary = %f”,NetSalary);  getch();  } |
| Algorithm | Step 1: Start  Step 2: Declare Net Salary, Basic Salary, HRA, DA, MA, TA, PF  Step 3: Read Basic Salary  Step 4: DA = 0.7 \* Basic Salary  HRA = 0.7 \* Basic Salary  MA = 0.7 \* Basic Salary  TA = 0.7 \* Basic Salary  PF = 0.12 \* Basic Salary  Net Salary = 0.12 \* Basic Salary + DA + HRA + MA + TA – PF  Step 5: Print Net Salary, DA, HRA, MA, TA, PF  Step 6: Stop |
| Flowchart | Print Net Salary, DA, HRA, MA, TA, PF  Read Basic Salary  DA – 0.7 \* Basic Salary , HRA – 0.07 \* Basic Salary ,  MA – 0,02 \* Basic Salary , TA – 0.04 \* Basic Salary ,  PF – 0.12 \* Basic Salary , Net Salary – Basic Salary + DA + HRA + MA + TA - PF  Declare Net Salary, Basic Salary, HRA,DA,MA,TA,PF |
| Output |  |
| 2.2 | The distance between two cities (in KMS) is input through the keyboard. Write a program to convert and print its distance in meters, feet, inches and centimeters. |
| Code (INPUT) | #include<stdio.h>  int main()  {  float km, meter, cm, feet, inches;  printf("Enter distance in kilometers: ");  scanf("%f", &km);  /\* calculate the conversion \*/  m = km \* 1000;  cm = km \* 1000 \* 100;  f = km \* 3280.84;  in = km \* 39370.08;  printf("The distance in Feet: %f\n", feet);  printf("The distance in Inches: %f\n", inches);  printf("The distance in Meters: %f\n", meter);  printf("The distance in Centimeters: %f\n", cm);  return (0);  } |
| Algorithm | Step 1: Start  Step 2: Declare variables (km, m, ft, inch, cm)  Step 3: Enter distance between two cities in kilometres  Step 4: Read value of given input in kilometres  Step 5: metres = km \* (1000);  feet = km \* (3280.84);  inches = km \* (39370);  cm = km \* (100000);  Step 6: Print the calculated distance in metre, cm, feet, inches  Step 7: Stop |
| Flowchart | Declare variables km, m, ft, inc, cm  Enter distance between two cities in km  Read value of given input in km  Meters = km \* (1000);  feet = km \* (3280.84);  inches = km \* (39370.1);  cm = km \* (100000);  Print the calculated distance in meter, cm, feet, inches. |
| Output | Text  Description automatically generated |
|  |  |
| 3.1 | Write a program to find the greatest of the three numbers entered through the keyboard using conditional operators. |
| Code(INPUT) | #include<stdio.h>  int main()  {  int a,b,c,max;  printf(“\nEnter the number 1 : ”);  scanf(“%d”, &a);  printf(“\nEnter the number 2 : ”);  scanf(“%d”, &b);  printf(“\nEnter the number 3 : ”);  scanf(“%d”, &c) |
|  | if(a>b && a>c)  {  printf(“%d is the largest number”,a);  }    else if(b>c)  {  printf(“%d is the largest number”,a);  }    else  {  printf(“%d is the largest number”,c);  }  return 0;  } |
| Algorithm | Step 1: Start  Step 2: Declare three numbers and max as an integer(a,b,c,max)  Step 3: Read three numbers  Step 4: if a>b & a>c then a is the greatest  Step 5: if b>c & b>a then b is the greatest  Step 6: if c>b & c>a then c is the greatest  Step 7: Print the greatest number  Step 8: Stop |
| Flowchart | Declare three variables (a, b and c) and max as an integer  Read a, b and c  If (a>b && a>c)  Print a is the greatest  True  False |
|  | If b>c  Print b is the greatest    True  False  Print c is the greatest |
| Output | Text  Description automatically generated |
| 3.2 | Any year is input through the keyboard. Write a program to determine whether the year is a leap year or not. Use the logical operators && and ||. |
| Algorithm | Step 1: Start  Step 2: Declare year as an integer  Step 3: Read Year  Step 4: if ((year%100 == 0 && year%400 == 0) ||(year%100 != 0 && year%4 == 0)) then year is leap year  Step 5: else it isn’t leap year  Step 6: Print the statement  Step 7: Stop |
| Flowchart | If((year%100 == 0 && year%400 == 0)  || (year%100 != 0 && year%4 == 0))  True  False  Print year is not a leap year  Print year is a leap year  Read year  Declare year as an integer |
| Code(INPUT) | #include<stdio.h>  int main()  {  int year;  printf(“Enter the year: ”);  scanf(“%d”, &year);  if((year%100 == 0 && year%400 == 0) || (year%100 ! = 0 && year%4 == 0))  {  printf(“%d is leap year”,year);  }  else  {  printf(“%d isn’t leap year”,year);  }  return 0;  } |
| Output | Text  Description automatically generated |
| 4.1 | Write a program to convert the decimal number into octal and hexadecimel format. |
| Algorithm | Step 1: Start  Step 2: Enter a decimal  Step 3: Read a decimal  Step 4: Convert decimal (%d) to octal (%o) and hexadecimal(%x)  Step 5: Stop |
| Flowchart | Print decimal (%d) and hexadecimal (%x)  Read a decimel  Enter a decimel  Convert decimal (%d) to octal (%o) and hexadecimal (%x) |
| Code(INPUT) | #include<stdio.h>  int main()  {  int n;  printf(“Enter a number (Decimal) :”);  scanf(“%d”,&n);    printf(“Octal number of %d : %o\n”,n,n);  printf(“Hexadecimal number of %d : %x\n”,n,n);  return 0;  } |
| Output |  |
| 4.2 | Write a C program to print multiplication table of number entered by user. |
| Algorithm | Step 1: Start  Step 2: Declare a as an integer  Step 3: read a  Step 4: Print a\*1  Step 5: Print a\*2  Step 6: Print a\*3  Step 7: Print a\*4  Step 8: Print a\*5  Step 9: Print a\*6  Step 10: Print a\*7  Step 11: Print a\*8  Step 12: Print a\*9  Step 13: Print a\*10  Step 14: Stop |
| Flowchart | Declare a as an integer  Read a |
|  | Print a\*10  Print a\*9  Print a\*8  Print a\*7  Print a\*6  Print a\*5  Print a\*1  Print a\*3  Print a\*4  Print a\*2 |
| Code (INPUT) | #include<stdio.h>  int main  {  int a;  printf(“Enter a number whose multiplication table you want to print : ”);  scanf(“%d”,&a);  printf(“multiplication tble:\n”);  printf(“%d \* 1 = %d\n”,a,a\*1);  printf(“%d \* 2 = %d\n”,a,a\*2);  printf(“%d \* 3 = %d\n”,a,a\*3);  printf(“%d \* 4 = %d\n”,a,a\*4);  printf(“%d \* 5 = %d\n”,a,a\*5);  printf(“%d \* 6 = %d\n”,a,a\*6);  printf(“%d \* 7 = %d\n”,a,a\*7);  printf(“%d \* 8 = %d\n”,a,a\*8);  printf(“%d \* 9 = %d\n”,a,a\*9);  printf(“%d \* 10 =%d\n”,a,a\*10);  return 0;  } |
| Output |  |
| 5.1 | If the cost price and selling price of an item is input through the keyboard, write a program to determine whether  the seller has made profit or incured loss. Also determine how much profit he made or how much loss he incured. |
| Algorithm | Step 1: Start  Step 2: Declare cost price, selling price and difference as float.  Step 3: Read cost price and selling price.  Step 4: if costprice < selling price then find difference <- costprice – sellingprice  Step 5: else selling price < cost price then find difference<- cost price – selling price  Step 6: print difference  Step 7: Stop |
| Flowchart | Difference = Cost price – selling price  Print difference  Difference = Selling price – Cost price  Cost price<selling price  Read cost price and selling price  Declare cost price, selling price and difference as float |
| Code (INPUT) | #include<stdio.h>  int main()  {  float cp,sp,diff;  printf(“Enter the cost price of your product: ”);  scanf(“%f”,&cp);  printf(“Enter the selling price of the product: ”);  scanf(“%f”,&sp);    if(cp<sp)  {  diff= sp-cp;  printf(“The seller has taken profit of %f bucks”,diff);  }  else  {  diff=cp-sp;  printf(“The seller has incured loss of %f bucks”,diff);  }    return 0;  } |
| Output | Text  Description automatically generated |
| 5.2 | If the ages of Ram, Shyam and Ajay are input through the keyboard, write a program to determine the youngest  of the three. |
| Algorithm | Step 1: Start  Step 2: declare ages of Ram, Shyam and Ajay as integer  Step 3: Read ages of Ram, Shyam and Ajay.  Step 4: Switch (Ram<Shyam && Ram<Ajay)  Step 5: Default:  Switch(Shyam < Ajay)  Case 1 : Print Shyam is youngest  Step 6: Default :  Ajay is youngest  Step 7: Stop |
| Code (INPUT) | #include<stdio.h>  int main()  {  int Ram,Shyam,Ajay;  printf(“Enter the age of Ram : ”);  scanf(“%d”,&Ram);  printf(“\nEnter the age of Shyam : ”);  scanf(“%d”,&Shyam);  printf(“\nEnter the age of Ajay : ”);  scanf(“%d”,&Ajay);  switch(Ram<Shyam && Ram<Ajay)  {  case 1:  printf(“Ram is the youngest”);  break;  default:  switch(Shyam<Ajay)  {  case 1:  printf(“Shyam is the youngest”);  break;  default:  printf(“Ajay is the youngest”);  }  }  return 0; |
|  | Text  Description automatically generated |
| Flowchart | Default  Case 1  Default  Case 1  Switch Shyam < Ajay  Read age of ram, shyam and ajay  Switch (Ram < Shyam && Ram < Ajay)  Ajay is the youngest  Ram is youngest  Shyam is the youngest  Declare ages of ram, shyam and ajay as integer |
| 5.3 | The policy followed by a company to process customer orders is given by  the following rules:  a) If a customer order is less than or equal to that in stock and ‘has  credit’ is OK, supply ‘has requirements’.  b) If ‘has credit’ is not OK do not supply. Send him intimation.  c) If ‘has credit’ is OK but the item in stock is less than ‘has ordered’,  supply what is in stock and Intimate him that the balance will be refunded.  Write a C program to implement the company policy. |
| Algorithm | Step 1 : Start  Step 2 : Declare order, price , requirement and stock as integer.  Step 3 : Read requirement and order  Step 4 : Stock←15  price←2500  Step 5 : If(requirement<Stock) then go to Step 6 otherwise goto step 8  Step 6 : if (order>price) then print your order is successfully done otherwise goto step 7.  Step 7 : print you haven’t enough money to buy it.  Step 8 : print out of stock  Step 9 : Stop |
|  | Flowchart   |  |  | | --- | --- | | Flowchart | false  You haven’t enough money to buy it  true  Your order is successfully done  Order > price  Out of stock  false  true  Requiremnt<stock  Stock←15  price←2500  Read requirement and order  Declare order, price , requirement and stock as integer. | | Program | #include<stdio.h>  int main()  {  int order,price,requirement,stock;  printf(“Enter the amount of the thing you want to purchase : ”);  scanf(“%d”,&requirement);    printf(“Enter the total price you have : ”);  scanf(“%d”,&order);  stock = 15;  price = 2500;  if(requirement<stock)  {  if(order>price)  {  printf(“your order is successfully done”);  }  else  {  printf(“You do not have enough money to buy it”);  }  }  else  {  printf(“out of stock”);  }    return 0;  } | |  | Text  Description automatically generated | | 6.1 | Two numbers are entered through the keyboard. Write a program to find the value of one number raised to the power of another. | | Algorithm | Step 1 : Start  Step 2 : Declare three variables i,j and a ←1.  Step 3 : Read the value of i and j.  Step 4 : a←a\*i  j←j – 1  Step 5 : if j←0 then go to next step otherwise go to step 4.  Step 6 : Print a  Step 7 : Stop | | Flowchart | True  Print a  false  a=a\*i  j=j – 1  j=0  Read the value of i and j  Declare three variables I,j and a←1 | | Program | #include<stdio.h>  int main()  {  int i,j,a=1;  printf(“Enter a number whose power of you want to : ”);  scanf(“%d”,&i);  printf(“Enter a numbe of power : ”);  scanf(“%d”,&j);  while(j!=0)  {  a = a\*1;  j = j – 1;  }  printf(“and = %d”,a);  return 0;  } | | Output | Graphical user interface, text  Description automatically generated | | 6.2 | Write a program to print the multiplication table of the number entered from the keyboard.  The table should get displayed in the following form:  12\*1=12  12\*2=24....  (Use for loop) | | Flowchart | true  false  c = a\*b  b++  Print a\*b=c  b = 11  Read a  declare a,b=1 and c as integer. | | Algorithm | Step 1 : Start  Step 2 : declare a,b←1 and c as integer.  Step 3 : Read a  Step 4 : c←a\*b  b++  Step 5 : print a\*b ← c  Step 6 : if b=11 then go to next step otherwise go to step 4.  Step 7 : Stop | | Program | #include<stdio.h>  int main()  {  int a,b,c;  printf(“Enter a number whose multiplication you want to do : ”);  scanf(“%d”,&a);  for(b=1;b<11;b++)  {  c = a \* b;  printf(“%d \* %2d = %2d\n”,a,b,c);  }  return 0;  } | | Output | A screen shot of a computer  Description automatically generated with low confidence | |

|  |  |
| --- | --- |
| 6.3 | Write a menu driven program which has following option:   1. Prime or not 2. Perfect number or not 3. Factorial of a number 4. Exit |
| Algorithm | Step 1: Start  Step 2 : Declare variables num,res,n,i,rem,flag←0 ,sum←0,c←0  Step 3 : print 1. Prime or not  2. Perfect number or not  3. Factorial of a number  4. Exit  Step 4 : read the choice of user(c)  Step 5 : switch(c) choose the case  Step 6 : if case←1(c←1) then go to Step 10  Step 7 : if case←2(c←2) then go to step 18  Step 8 : if case←3(c←3) then go to step 28  Step 9 : if case←4(c←4) then go to step 37  Step 10 : Case 1 : Read the value of integer entered by user(num)  Step 11 : n ← num  Step 12 : res ← num  Step 13 : res ←res\*(num-1)  Step 14 : num ← num-1  Step 15 : if num ← 1 then go to next step else go to step 11  Step 16 : print the factorial of number(res)  Step 17 : break the statement  Step 18 : case 2 : Read the value of integer entered by user(num)  Step 19 : n ← num  Step 20 : i ← 2  Step 21 : i++  Step 22 : if num%i ← 1 then break the statement with flag← 1  Step 23 : if i>n/2 then go to next step else go to step 19  Step 24 : if num←1 then print 1 is neither prime nor composite else go to next step  Step 25 : if flag ←0 then print that the number is prime number else go to next step  Step 26 : print that the number is not a prime number  Step 27 : break the statement  Step 28 : case 3 : Read the value of integer entered by user(num)  Step 29 : i ←1  Step 30 : rem←num%i  Step 31 : if rem←0 then sum←sum+i  Step 32 : i++  Step 33 : if i<num then go to step 30 else go to next step  Step 34 : if sum←num then print it is a perfect number else go to next step  Step 35 : print it is not a perfect number  Step 36 : break the statement  Step 37 : case 4 : print exit  Step 38 : break the statement  Step 39 : if c←4 then go to step 37 else go to step 3.  Step 40 : stop. |
| Code (INPUT) | #include<stdio.h>  #include<conio.h>  int main()  {  int c=0, num, res, n, flag=0, i,sum=0,rem;  while(c!=4)  {  printf("\n1. Factorial of a number\n2. Prime  or not\n3. perfect or not\n4. Exit\n");  printf("\nEnter your choice:");  scanf("%d", &c);  switch(c)  {  case 1:  printf("Enter an integer: ");  scanf("%d", &num);  n=num;  res=num;  while(num>1)  {  res = res\*(num-1);  num = num-1;  }  printf("\nFactorial of %d is %d. \n\n",n, res);  break;  case 2:  printf("Enter an integer: ");  scanf("%d", &num);  n=num;  for(i=2;i<=n/2;i++)  {  if(num%i==0)  {  flag=1;  break;  }  }  if(num==1)  printf("\n1 is neither prime nor  composite");  else  {  if(flag==0)  printf("\n%d is Prime  Number.\n\n", n);  else  printf("\n%d is not a  Prime Number.\n\n", n);  }  break;  case 3:  printf("Enter a number\n");  scanf("%d", &num);    for(i = 1; i < num; i++)  {  rem = num % i;  if (rem == 0)  {  sum = sum + i;  }  }  if (sum == num)  {  printf("it is a Perfect Number");  }  else  {  printf("\n it is not a Perfect  Number");  }  break;  case 4:  printf("\nExit");  break;  }  }  } |
| Output |  |
| 6.4 | Write a program for a match stick game between the computer and a user. Your program should ensure that the computer always wins.  Rules for the game are as follows:   * There are 21 match sticks * The computer asks the player to pick 1,2,3 or 4 match sticks. * After the person picks, the computer does its picking. * Whoever is forced to pickup the last matchstick loses |
| Algorithm | Step 1 : Start  Step 2 : Declare variables m←21,comp and num.  Step 3 : print instructions(rules)  Step 4 : read the number of match sticks user want to remove(num)  Step 5 : m -= num  Step 6 : print m  Step 7 : comp ← 5-m  Step 8 : print comp  Step 9 : m -= comp  Step 10 : print m  Step 11 : if m←1 then go to next step else go to step 4.  Step 12 : print you lose the game  Step 13 : Stop |
| Flowchart |  |

|  |
| --- |
| A  m-=comp  Print comp  comp ← 5-m  Print m  m -= num  Read the number of match sticks user want to remove(num)  Print instructions  Declare variables m←21,comp and num |

|  |
| --- |
| false  true  Print you lose the game  m←1  Print m  A |
| **Program**:  #include<stdio.h>  void main()  {  int m=21,num,comp;    printf("There are 21 match sticks.\n");  printf("The rules for the game are as follows: \n1.User can only take 1 or 2 or 3 or 4 match sticks.");  printf("\n2.First user will take out match sticks.\n3.The one who takes out the last match sticks loses.\n");  while(m!=1)  {  printf("Enter the no. of match sticks you want to remove : ");  scanf("%d",&num);  m-=num;  printf("No. of match stick left: %d\n",m);  comp=5-num;  printf("The no. of match sticks taken out by  computer is: %d\n",comp);  m-=comp;  printf("No. of match stick left: %d\n",m);  }  printf("You lost the game\n");  } |
| **Output:** |