|  |  |
| --- | --- |
| **Name** | **PRATIK PUJARI** |
| **UID no.** | **2020300054** |
| **Experiment No.** | 6 |

|  |  |
| --- | --- |
| **AIM:** | Apply the concept of recursion to solve a problem. |
| **Program 1** | |
| **PROBLEM STATEMENT :** | Given a number n, print following a pattern without using any loop |
| **PROGRAM:** | ALGORITHM:  void main()  STEP 1: START.  STEP 2: Initialize the variable “num” and input the number from the user and store it in the variable.  STEP 3: Printf(“The pattern is:”)  STEP 4: Call the predefined function print\_pat(num).  STEP 5: END.  print\_pat1(num)  STEP 1: START.  STEP 2: Initialize a static variable “count” and set it to zero.  STEP 3: Printf the num.  STEP 4: If (num>0) ,do print\_pat2(num+5,count),else go to step 5.  STEP 5: Do count++ and return printpat1(num-5).  STEP 6: END.  int print\_pat2(int num, int count)  STEP 1: START.  STEP 2: If count is greater than zero execute step 3 and 4 or else go to step 5.  STEP 3: Do printf the value of num and decrement the value of counter by one.  STEP 4: Return print\_pat2(num+5,count).  STEP 5: Return 0.  PROGRAM:  #include<stdio.h>  int print\_pat1(int);  int print\_pat2(int,int);  void main()  {  int num;  printf("Enter the number:");  scanf("%d",&num);  printf("The pattern of the %d is:",num);  print\_pat1(num);  }  int print\_pat1(int num)  {  int static count=0;  printf("%d ",num);  if(num<0)  {  print\_pat2(num+5,count);  }  else  {  count++;  return print\_pat1(num-5);  }  }  int print\_pat2(int num,int count)  {  if(count>0)  {  printf("%d ",num);  count--;  return print\_pat2(num+5,count);  }  else  {  return 0;  }  } |
| **RESULT: The number is printed according to the given pattern.** | |
| **INPUT:** | 69 |
| **OUTPUT:** | Enter the number:69  The pattern of the 69 is:69 64 59 54 49 44 39 34 29 24 19 14 9 4 -1 4 9 14 19 24 29 34 39 44 49 54 59 64 69 |

|  |  |
| --- | --- |
| **Program 2** | |
| **PROBLEM STATEMENT :** | Write a function which given m and n returns A(m,n). Tabulate the values of A(m,n) for all  m in the range 1 to 4 and all n in the range 1 to 10. |
| **PROGRAM:** | ALGORITHM:  void main()  STEP 1: START.  STEP 2: Initialize the u1, l1,u2,l2 to find the ranges of the ackerman number.  STEP 3: Input the range of the first value and store it in u1,l1.  STEP 4: Input the range of the second value and store it in u2,l2.  STEP 5: Printf(“The ackerman values of the numbers are:”).  STEP 6: For i=l1 and less than or equal to u1, Repeat the steps 6.1 and 6.2 or else if the condition fails go to step 7.  STEP 6.1: For j=l2 and less than equal to u2, Repeat the steps 6.1.1 and 6.1.2 or else if the condition fails go to step 6.2.  STEP 6.1.1: Printf(“A(%d,%d)= %d \n”,i,j,find\_ack(i,j).  STEP 6.1.2: Increment the loop counter j by one.  STEP 6.2: Increment the loop counter i by one.  STEP 7: END.  int find\_ack(int n1,int n2)  STEP 1: START.  STEP 2: If n1 is equal to zero return n2+1 or else go to step 3.  STEP 3: If n2 is equal to zero return find\_ack(n1-1,1) or else go to step 4.  STEP 4: If n1 and n2 not equal to zero return find\_ack(n1-1,find(n1,n2-1))  STEP 5: END.  PROGRAM:  #include<stdio.h>  int u1,l1,u2,l2;  int find\_ack(int,int);  int ack2(int ,int);  void main()  {  printf("Enter the range of the first value:");  scanf("%d %d",&l1,&u1);  printf("Enter the range of the second value:");  scanf("%d %d",&l2,&u2);  printf("The Ackerman's values of the number are given below:\n");  for(int i=l1;i<=u1;i++)  {  for(int j=l2;j<=u2;j++)  {  printf("A(%d,%d)=%d \n",i,j,find\_ack(i,j));  }  }  }  int find\_ack(int n1,int n2)  {  if(n1==0)  {  return (n2+1);  }  else if(n2==0)  {  return find\_ack(n1-1,1);  }  else if(n1!=0 && n2!=0)  {  return find\_ack(n1-1,find\_ack(n1,n2-1));  }  } |
| **RESULT: All the Ackerman numbers in the range given by the user are printed.** | |
| **INPUT:** | 1 3  1 10 |
| **OUTPUT:** | Enter the range of the first value:1 3  Enter the range of the second value:1 10  The Ackerman's values of the number are given below:  A(1,1)=3  A(1,2)=4  A(1,3)=5  A(1,4)=6  A(1,5)=7  A(1,6)=8  A(1,7)=9  A(1,8)=10  A(1,9)=11  A(1,10)=12  A(2,1)=5  A(2,2)=7  A(2,3)=9  A(2,4)=11  A(2,5)=13  A(2,6)=15  A(2,7)=17  A(2,8)=19  A(2,9)=21  A(2,10)=23  A(3,1)=13  A(3,2)=29  A(3,3)=61  A(3,4)=125  A(3,5)=253  A(3,6)=509  A(3,7)=1021  A(3,8)=2045  A(3,9)=4093  A(3,10)=8189 |