

PDCAP14 - PROGRAMMING IN C LAB

RECORD PROGRAMS

Ex. No. 1 Date :	GIVEN NUMBER IS ODD OR EVEN USING SIMPLE IF-ELSE
---------------------	--

AIM: To write a C program to check the given number is odd or even using simple if-else.

ALGORITHM:

- Step 1 : Start the Process.
- Step 2 : Include the proper header files.
- Step 3 : Declare the variable a.
- Step 4 : Get the value of the variable a.
- Step 5 : If a modular by 2, the remainder equal to 0
Then Print "The given number is EVEN.
Otherwise print "The given number is ODD.
- Step 6 : Terminate the process.

PROGRAM:

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

int main()
{
int a;
clrscr();
printf("\nEnter a: ");
scanf("%d", &a);

//logic
if (a % 2 == 0) {
printf("\nThe given number is EVEN");
}
else {
printf("\nThe given number is ODD");
}
getch();
return 0;
}
```

OUTPUT:

Enter a:2
The given number is EVEN

Enter a:3
The given number is ODD

RESULT:

Thus the program was successfully executed and verified.

Ex. No. 2	LEAP YEAR OR NOT USING ELSE-IF-LADDER
Date :	

AIM: To write a C program to check the given year is leap year or not using else-if-ladder.

ALGORITHM:

Step 1 : Start the process.
Step 2 : Include the appropriate header files
Step 3 : Declare the variable year.
Step 4 : If year modular by 400, the remainder equal to zero
Then print the given year is a leap year.

Else if year modular by 100, the remainder equal to zero
Then print the given year is no leap year.

Else if year modular by 4, the remainder equal to zero
Then print the given year is a leap year.

Otherwise the year is not a leap year.
Step 5: Terminate the process.

PROGRAM:

```
#include <stdio.h>
#include <conio.h>
void main()
{
int year;
clrscr();
printf("\nInput a year :");
scanf("%d",&year);
if((year %400)==0)
printf("%d is a leap year.\n",year);
else if((year %100)==0)
printf("%d is a not leap year.\n",year);
else if((year %4)==0)
printf("%d is a leap year.\n",year);
else
```

```
printf("%d is not a leap year \n",year);
```

```
getch();  
}
```

OUTPUT:

```
Input a year: 2023  
2023 is not a leap year
```

```
Input a year: 2024  
2024 is a leap year
```

RESULT:

Thus the program was successfully executed and verified.

Ex. No. 3 Date :	LARGEST NUMBER OUT OF THE THREE GIVEN NUMBERS USING IF-ELSE
---------------------	---

AIM: To write a C program to find the maximum number out of the three given numbers using if-else statement.

ALGORITHM:

Step 1 : Start the process.
Step 2 : Include the appropriate header files.
Step 3 : Declare the variable A,B and C.
Step 4 : Get the value of A,B and C.
Step 5 : If A is greater than or equal to B and A is greater than Or equal to C then print "A is the largest number".

Else if B is greater than or equal to A and B is greater than or equal to C then print "B is the largest number".

Otherwise, print "C is the largest number".

Step 6 : Terminate the process.

PROGRAM:

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

```
Int main()
```

```
{
```

```
int A, B, C;
```

```
clrscr();
```

```
printf("\nEnter the numbers A, B and C: ");
```

```
scanf("%d %d %d", &A, &B, &C);
```

```
// finding max using compound expressions
```

```

if(A >= B && A >= C)
    printf("%d is the largest number.", A);

else if(B >= A && B >= C)
    printf("%d is the largest number.", B);

else
    printf("%d is the largest number.", C);

    getch();
    return(0);
}

```

OUTPUT:

```

Enter the numbers A, B and C: 7 12 34
34 is the largest number.

```

RESULT:

Thus the program was successfully executed and verified.

Ex. No. 3A	LARGEST NUMBER AMONG THREE NUMBERS USING NESTED IF-ELSE
Date :	

AIM: To write a C program to find the largest number among three numbers using nested if-else.

ALGORITHM:

Step 1 : Start the process.

Step 2 : Include the appropriate header files.

Step 3 : Declare the variable A,B and C.

Step 4 : Get the value of A,B and C.

Step 5 : If A is greater than or equal to B then if A is greater than or equal to C then print "A is the largest number", Otherwise print "C is the largest number".

Otherwise, if B is greater than or equal to C then print "B is the largest number", Otherwise print "C is the largest number".

Step 6 : Terminate the process.

PROGRAM:

```

#include <stdio.h>
#include <conio.h>
int main()
{
    int A, B, C;
    clrscr();
    printf("Enter three numbers: ");

```

```

scanf("%d %d %d", &A, &B, &C);
if(A >= B) {
    if(A >= C)
        printf("%d is the largest number.", A);
    else
        printf("%d is the largest number.", C);
}
else{
    if(B >= C)
        printf("%d is the largest number.", B);
    else
        printf("%d is the largest number.", C);
}
getch();

return(0);

```

OUTPUT:

```

Enter three numbers: 11 34 5
34 is the largest number.

```

RESULT:

Thus the program was successfully executed and verified.

Ex. No. 4 Date :	CONVERT DIGIT TO WORD USING SWITCH STATEMENT
---------------------	--

AIM: To write a C program to convert digit to word using switch statement.

ALGORITHM:

- Step 1 : Start the process.
- Step 2 : Include the appropriate header files.
- Step 3 : Declare the variable digit.
- Step 4 : Get the value of digit.
- Step 5 : The switch variable is evaluated.
- Step 6 : The evaluated value is matched against all the present cases.
- Step 6A: If the matching case value is found, the associated code is executed.
- Step 6B: If the matching code is not found, then the default case is executed if present.
- Step 7 : Terminated the process.

PROGRAM:

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

```

#include <conio.h>
int main()
{
int digit;
clrscr();
printf("\nEnter digit:");
scanf("%d", &digit); //Input digit

switch (digit)
{
//Writing a case for every digit in 0-9

case 1:
printf("One\n");
break;
case 2:
printf("Two\n");
break;
case 3:
printf("Three\n");
break;
case 4:
printf("Four\n");
break;
case 5:
printf("Five\n");
break;
case 6:
printf("Six\n");
break;
case 7:
printf("Seven\n");
break;
case 8:
printf("Eight\n");
break;
case 9:
printf("Nine\n");
break;
case 0:
printf("Zero\n");
break;
default:
printf("Invalid Digit\n");
}
getch();
return 0;
}

```

OUTPUT:

Enter digit: 9
Nine

Enter digit: 10
Invalid Digit

RESULT:

Thus the program was successfully executed and verified.

Ex. No. 5 Date :	REVERSE THE GIVEN NUMBER USING WHILE LOOP
---------------------	---

AIM: To write a C program to reverse the given number using while loop.

ALGORITHM:

Step 1 : Start the process.
Step 2 : Include the appropriate header files.
Step 3 : Declare the variable n, rev and rem.
Step 4 : Get the value of n.
Step 5 : While loop statements (step 6 to 8) will be executed until the n value is not equal to 0

Step 6 : n modular by 10, remainder value store in rem.
Step 7 : rev value multiple by 10 then add to rem, final value store in rev.
Step 8 : n divide by 10, then assign the value to n.
Step 9 : Print the reverse number.
Step 10: Terminate the process.

PROGRAM:

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

int main()
{

int n, rev = 0, rem;
clrscr();
printf("\nEnter an integer: ");
scanf("%d", &n);

while (n != 0) {
    rem = n % 10;
    rev = rev * 10 + rem;
    n /= 10;
}
```

```
printf("\nReversed number = %d", rev);
getch();
return 0;
}
```

OUTPUT:

```
Enter an integer: 2345
Reversed number = 5432
```

RESULT:

Thus the program was successfully executed and verified.

Ex. No. 6	ARMSTRONG NUMBER OR NOT USING DO-WHILE LOOP
Date :	

AIM: To write a C program to check the given number is Armstrong number or not using do while loop.

ALGORITHM:

- Step 1 : Start the process.
- Step 2 : Include the appropriate header files.
- Step 3 : Declare the variable n, num, r and ans=0;
- Step 4 : Get the value of num.
- Step 5 : Assign the value num to n.
- Step 6 : Do-while loop statements (step 7 to 9) will be executed until the n value is greater than 0.
- Step 7 : n modular by 10, the remainder assign to r.
- Step 8 : cube the value of r, then add to ans finally value assign to the ans.
- Step 9 : n divide by 10, then assign the value to n.
- Step 10: If check the value of variable ans and num, both are equal then print " the given number is an Armstrong number".
- Otherwise, print "the given number is not an Armstrong number".
- Step 10: Terminate the process.

PROGRAM:

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

void main()
{
int n,num,r,ans=0;
```

```

clrscr();

printf("\nEnter a positive integer: ");
scanf("%d", &num);
n=num;

/* Loop to calculate the sum of the cubes of its digits */
do
{
    r=n%10;
    ans=ans+r*r*r;
    n=n/10;
}while(n>0);

/* if else condition to print Armstrong or Not */
if(ans==num)
{
    printf("%d is an Armstrong number.",num);
}
else
{
    printf("%d is not an Armstrong number.",num);
}
getch();
}

```

OUTPUT:

```

Enter a positive integer: 123
123 is not an Armstrong number.

```

```

Enter a positive integer: 153
153 is an Armstrong number.

```

RESULT:

Thus the program was successfully executed and verified.

Ex. No. 7 Date :	ASCENDING ORDER USING ONE DIMENSIONAL ARRAY AND NESTED FOR LOOP
---------------------	--

AIM: To write a C program to sort the given numbers in ascending order using array and nested for loop.

ALGORITHM:

- Step 1 : Start the process.
- Step 2 : Include the appropriate header files.
- Step 3 : Declare the variable n,i,j,tmp and array variable arr1[100];

Step 4 : Get the size of the array n.

Step 5 : Get the n number of array element value arr1[i] using for loop.

Step 6 : Sorted the array elements using nested for loop and arrange the values in ascending order.

```
        if(arr1[j]<arr1[i])
        {
            tmp= arr1[i];
            arr1[i]= arr1[j];
            arr1[j]=tmp;
        }
```

Step 7 : Print the array element in ascending order using for loop.

Step 8 : Terminated the process.

PROGRAM:

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

```
void main()
```

```
{
int arr1[100];
int n,i, j,tmp;
```

```
clrscr();
```

```
printf("\n\nSort elements of array in ascending order :\n ");
printf("-----\n");
```

```
printf("\nInput the size of array : ");
scanf("%d", &n);
```

```
printf("\nInput %d elements in the array :\n",n);
```

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

```
{
    printf("\nElement - %d : ",i+1);
    scanf("%d",&arr1[i]);
}
```

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

```
{
for(j=i+1; j<n;j++)
```

```
{
if(arr1[j]<arr1[i])
```

```
{
            tmp= arr1[i];
            arr1[i]= arr1[j];
            arr1[j]=tmp;
```

```
    }
}
```

```
printf("\nElements of array in sorted ascending order:\n");
```

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

```
{
```

```

printf("%d  ", arr1[i]);
}
    printf("\n\n");
    getch();
}

```

OUTPUT:

Sort elements of array in ascending order

Input the size of array :4

Input 4 elements in the array :

```

Element - 1 :12
Element - 2 :10
Element - 3 :25
Element - 4 :22

```

Elements of array in sorted ascending order:

10 12 22 25

RESULT:

Thus the program was successfully executed and verified.

Ex. No. 8 Date :	STUDENT MARK STATEMENT USING ARRAY OF STRUCTURES
---------------------	---

AIM : To write a C Program to prepare the student mark statement using array of structures

ALGORITHM:

- Step 1: Start the process.
- Step 2: Include the appropriate header files.
- Step 3: Create the structure variable student with name, rollno and marks.
- Step 4: Declare the variable i,n.
- Step 5: Declare the size of structure variable student.
- Step 6: Get the value of n.
- Step 7: Get the values of name,rollno and marks for n number of Students using for loop.
- Step 8: Print the name, rollno and marks for n number of students using for loop.
- Step 9: Terminate the process.

```

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

struct student {
    char name[20];
    int rollno;
    float marks;
};

int main( )
{
    int i,n;
    clrscr();
    printf("\nEnter how many records u want to store :: ");
    scanf("%d",&n);
    struct student stuarr[n];
    printf("\nEnter name, roll no. and marks Below :: \n");

    for(i=1; i<=n; i++)
    {
        printf("\nEnter %d record :: \n",i);

        printf("\nEnter Name :: ");
        scanf("%s",stuarr[i].name);
        printf("\nEnter RollNo. :: ");
        scanf("%d",&stuarr[i].rollno);
        printf("\nEnter Marks :: ");
        scanf("%f",&stuarr[i].marks);

    }
    printf("\n\tName\tRollNo\tMarks\t\n");
    for(i=1; i<=n; i++)
        printf("\t%s\t%d\t%.2f\t\n", stuarr[i].name, stuarr[i].rollno,
        stuarr[i].marks);
    getch();
    return 0;
}

```

OUTPUT:

Enter how many records u want to store :: 5

Enter name, roll no. and marks Below ::

Enter 1 record ::

Enter Name :: Kannan

Enter RollNo. :: 101

Enter Marks :: 67

Enter 2 record ::

Enter Name :: Shyam

Enter RollNo. :: 201

Enter Marks :: 88.75

Enter 3 record ::

Enter Name :: Vimala
Enter RollNo. :: 301
Enter Marks :: 55.68

Enter 4 record ::
Enter Name :: Ramasamy
Enter RollNo. :: 401
Enter Marks :: 77.50

Enter 5 record ::
Enter Name :: Saranya
Enter RollNo. :: 501
Enter Marks :: 99

Name	RollNo	Marks
Kannan	101	67.00
Shyam	201	88.75
Vimala	301	55.68
Ramasamy	401	77.50
Saranya	501	99.00

RESULT:

Thus the program was successfully executed and verified.

Ex. No. 9 Date :	MATRIX ADDITION USING TWO DIMENSIONAL ARRAY
---------------------	---

AIM: To write a C program to find the sum of two matrices of order 2*2.

ALGORITHM:

- Step 1: Start the process.
- Step 2: Include the appropriate header files.
- Step 3: Declare the two dimensional array variable a[2][2], b[2][2] and result[2][2].
- Step 4: Get the value of first matrix elements using nested for loop.
- Step 5: Get the value of second matrix elements using nested for loop.
- Step 6: Print the resultant matrix values.
- Step 7: Terminate the process.

PROGRAM:

```
#include <stdio.h>
#include <conio.h>
int main()
{
```

```

float a[2][2], b[2][2], result[2][2];
int i,j;
// Taking input using nested for loop
clrscr();
printf("\nEnter elements of 1st matrix\n");
for (i = 0; i < 2; ++i)
    for (j = 0; j < 2; ++j)
    {
printf("\nEnter a(%d,%d): ", i, j);
scanf("%f", &a[i][j]);
    }

// Taking input using nested for loop
printf("\nEnter elements of 2nd matrix\n");
for (i = 0; i < 2; ++i)
    for (j = 0; j < 2; ++j)
    {
printf("\nEnter b(%d,%d): ", i,j);
scanf("%f", &b[i][j]);
    }

// adding corresponding elements of two arrays
for (i = 0; i < 2; ++i)
    for (j = 0; j < 2; ++j)
    {
        result[i][j] = a[i][j] + b[i][j];
    }

// Displaying the sum
printf("\nSum Of Matrix:\n");

for (i = 0; i < 2; ++i)
    for (j = 0; j < 2; ++j)
    {
printf("%.1f\t", result[i][j]);

        if (j == 1)
printf("\n");
    }
getch();
return 0;
}

```

OUTPUT:

Enter elements of 1st matrix

Enter a(0,0): 2;

Enter a(0,1): 0.5;

Enter a(1,0): -1.1;

Enter a(1,1): 2;

Enter elements of 2nd matrix

Enter b(0,0): 0.2;

Enter b(0,1): 0;

Enter b(1,0): 0.23;
Enter b(1,1): 23;

Sum Of Matrix:
2.2 0.5
-0.9 25.0

RESULT:

Thus the program was successfully executed and verified.

Ex. No. 10 Date :	FIND FACTORIAL USING RECURSIVE FUNCTION
----------------------	---

AIM: To write a C program to find the factorial of given number using recursive function.

ALGORITHM:

Step 1: Start the Process.
Step 2: Include the appropriate header files.
Step 3: Declare and define the fact() function
 Declare the variable I, fact=1 and n.
 Get the value of n.
 Call fact() multiple the i value with fact until i value is less than or equal to n, finally return the fact value.
Step 4: Print the fact value.
Step 5. Terminate the process.

PROGRAM:

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

int main()
{
clrscr();
printf("Enter a Number to Find Factorial: ");
printf("\nFactorial of a Given Number is: %d ",fact());
getch();
return 0;
}
int fact()
{
int i,fact=1,n;
scanf("%d",&n);
for(i=1; i<=n; i++)
```

```

    {
        fact=fact*i;
    }
    return fact;
}

```

OUTPUT:

Enter a Number to Find Factorial: 5

Factorial of a Given Number is: 120

RESULT:

Thus the program was successfully executed and verified

Ex. No. 10A Date :	ARITHMETIC OPERATION USING FUNCTIONS
-----------------------	--------------------------------------

AIM: To write a C Program to Perform Arithmetic Operations Using Functions

ALGORITHM:

- Step 1: Start the process.
- Step 2: Include the appropriate header files.
- Step 3: Declare and define the addition(int,int)function, calculate and return the sum value.
- Step 4: Declare and define the subtract(int,int)function, calculate and return the difference value.
- Step 5: Declare and define the multiply(int,int)function, calculate and return the multiply value.
- Step 6: Declare and define the division(int,int)function, calculate and return the divide value.
- Step 7: Declare and define the mod(int,int)function, calculate and return the rem value.
- Step 8: Declare the num1 and num2.
- Step 9: Get the value of num1 and num2.
- Step 10: Call the addition()function then print the sum value.
- Step 11: Call the subtract()function then print the difference value.
- Step 12: Call the multiply()function then print the multiply value.
- Step 12: Call the division() function then print the divide value.
- Step 13: Call the mod()function then print the remainder value.
- Step 14: Terminate the process.

PROGRAM:

```

#include <stdio.h>
#include <conio.h>
// Declaring addition function

```

```

int addition(int a, int b)
{
int sum = a + b;
return sum;
}
// Declaring subtraction function
int subtract(int a, int b)
{
int difference = a - b;
return difference;
}
// Declaring multiplication function
int multiply(int a, int b)
{
int multiply = a * b;
return multiply;
}
// Declaring division function
float division(float a, float b)
{
float divide = a / b;
return divide;
}
// Declaring modulus function
float mod(int a, int b)
{
int rem = a % b;
return rem;
}
int main()
{
int num1, num2;
// Asking for input
clrscr();
printf("\nEnter the first number: ");
scanf("%d", &num1);
printf("\nEnter the second number: ");
scanf("%d", &num2);
// Displaying output
printf("\nArithmetic operations on %d and %d: \n", num1, num2);
printf("Addition: %d\n", addition(num1, num2));
printf("Subtraction: %d\n", subtract(num1, num2));
printf("Multiplication: %d\n", multiply(num1, num2));
printf("Division: %f\n", division(num1, num2));
printf("Modulus: %d\n", mod(num1, num2));
getch();
return 0;
}

```

OUTPUT:

```

Enter the first number: 8
Enter the second number: 3

```

Arithmetic operations on 8 and 3:

Addition: 11

Subtraction: 5

Multiplication: 24

Division: 2.666667

Modulus: 3

RESULT:

Thus the program was successfully executed and verified

Ex. No. 11	SWAP TWO VALUE USING CALL BY VALUE
Date :	

AIM: To write a C program to swap two value using call by value.

ALGORITHM:

Step 1: Start the process.

Step 2: Include the appropriate header files.

Step 3: Declare and assign the variable a=10 and b=20.

Step 4: Declare and define the function swap(int,int)
int t;t=x;x=y;y=t;

Step 5: Print the value a and b, before call the swap()function.

Step 6: Call the Swap() function.

Step 7: Print the value a and b, inside the swap function after swapping the values.

Step 8: Terminate the process.

PROGRAM:

```
# include<stdio.h>
# include<conio.h>
void main()
{
void swap(int,int);
int a,b; a=10; b=20;
clrscr();
printf("\nPrint a and b values inside main function before call
swap()\n");
printf("\nThe value of a before swapping=%d",a);
printf("\nThe value of b before swapping=%d",b);
swap(a,b);
printf("\nPrint a and b values inside main function after call
swap()\n");
printf("\nThe value of a after swapping=%d",a);
printf("\nThe value of b after swapping=%d",b);
getch();
}
void swap(int a, int b)
```

```

{
int t;
t=a;
a=b;
b=t;
printf("\nPrint a and b values inside swap() function\n");
printf("\nThe value of a after swapping=%d",a);
printf("\nThe value of b after swapping=%d",b);
}

```

OUTPUT:

```

Print a and b values inside main function before call swap()
The value of a before swapping=10
The value of b before swapping=20

```

```

Print a and b values inside swap() function
The value of a after swapping=20
The value of b after swapping=10

```

```

Print a and b values inside main function after call swap()
The value of a before swapping=10
The value of b before swapping=20

```

RESULT:

Thus the program was successfully executed and verified

<p>Ex. No. 12 Date :</p>	<p>STRING FUNCTIONS USING SWITCH CASE</p>
------------------------------	---

AIM: To Write a C program to accept a string and perform various operations:

1. To convert string into upper case.
2. To reverse the string.
3. To copy string into another string.
4. To compute length depending upon user choice.
5. Exit

ALGORITHM:

- Step 1: Start the process.
- Step 2: Include the appropriate header files.
- Step 3: Declare the string variables str, str1.
- Step 4: Declare the variable opt, len.
- Step 5: Display the menu items.
- Step 6: Get the string value str.
- Step 7: Get the value for opt. Evaluate the opt value and appropriate switch case will be executed.
 - Case 1: String convert to upper case and print the string.
 - Case 2: Reverse the given string and print.

Case 3: Copy the string to str1, then print the string.

Case 4: Compute the length of the string and print the len.

Case 5: Exit the program.

Step 8: Terminate the process.

PROGRAM:

```
# include<stdio.h>
# include<conio.h>
#include<string.h>
void main()
{
char str[20];
char str1[20];
int opt,len;
clrscr();
printf("\n MAIN MENU\n");
printf("\n 1. Convert string into upper case");
printf("\n 2. Reverse the string");
printf("\n 3. Copy one string into another string");
printf("\n 4. Compute length of string");
printf("\n 5. Exit");
printf("\nEnter string :");
scanf("%s", &str);
printf("\nEnter your choice :");
scanf("%d",&opt);
switch(opt)
{
case 1:
strupr(str);
printf("\nThe string in uppercase is :%s",str);
break;
case 2:
strrev(str);
printf("\nThe reverse of string is : %s",str);
break;
case 3:
strcpy(str1,str);
printf("\nNew copied string is : %s",str1);
break;
case 4:
len=strlen(str);
printf("\nThe length of the string is : %d",len);
break;
case 5:
exit(0);
default:
printf("\nYou have entered a wrong choice.");
}
getch();
}
```

OUTPUT:

MAIN MENU

1. Convert string into upper case
2. Reverse the string
3. Copy one string into another string
4. Compute length of string
5. Exit

Enter string :Kannan

Enter your choice :1

The string in uppercase is :KANNAN

MAIN MENU

1. Convert string into upper case
2. Reverse the string
3. Copy one string into another string
4. Compute length of string
5. Exit

Enter string :Kannan

Enter your choice :2

The reverse of string is :nannaK

MAIN MENU

1. Convert string into upper case
2. Reverse the string
3. Copy one string into another string
4. Compute length of string
5. Exit

Enter string :Kannan

Enter your choice :3

New copied string is :Kannan

MAIN MENU

1. Convert string into upper case
2. Reverse the string
3. Copy one string into another string
4. Compute length of string
5. Exit

Enter string :Kannan

Enter your choice :4

The length of the string is :6

RESULT:

Thus the program was successfully executed and verified

Ex. No. 13 Date :	CREATE EMPLOYEE'S RECORD USING STRUCTURE
----------------------	--

AIM: To write a C program to read and print employee's record using structure

ALGORITHM:

- Step 1: Start the process.
- Step 2: Declare and define the employee structure with variables name, empId and salary.
- Step 3: Declare the structure variable emp.
- Step 3: Get the values of structure variable emp.name, emp.empId and emp.salary.
- Step 4: Print the values of emp.name, emp.empId and emp.salary.
- Step 5: Terminate the process.

PROGRAM:

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

/*structure declaration*/
typedef struct employee{
    char    name[30];
    int     empId;
    float   salary;
};

int main()
{
    /*declare structure variable*/
    struct employee emp;

    /*read employee details*/
    clrscr();
    printf("\nEnter details :\n");
    printf("\nName ?:" );
    gets(emp.name);
    printf("\nID ?:" );
    scanf("%d", &emp.empId);
    printf("\nSalary ?:" );
    scanf("%f", &emp.salary);

    /*print employee details*/
    printf("\nEntered detail is:");
    printf("\nName: %s", emp.name);
    printf("\nId: %d", emp.empId);
    printf("\nSalary: %f\n", emp.salary);
    getch();
    return 0;
}
```

OUTPUT:

Enter details :

Name ? : Kavitha

ID ? : 1120

Salary ? : 76543

Entered detail is:

Name: Kavitha

Id: 1120

Salary: 76543.000000

RESULT:

Thus the program was successfully executed and verified

Ex. No. 14	MEMORY ALLOCATION FOR UNION AND STRUCTURE
Date :	

AIM: To write a C program to find difference between union and structure for memory allocation.

ALGORITHM:

Step 1: Start the Process.

Step 2: Include the appropriate header files.

Step 3: Declare and define the union with variable name, salary, workerno.

Step 4: Declare and define the structure with variables name, salary, workerno.

Step 5: Create the structure and union variable sJob, uJob.

Step 6: Calculate the memory size of variable sJob, uJob with sizeof() operator and then print.

Step 7: Terminate the process.

PROGRAM:

```
#include <stdio.h>
#include <conio.h>
union unionJob
{
    //defining a union
    char name[32];
    float salary;
    int workerNo;
} uJob;

struct structJob
{
    char name[32];
    float salary;
    int workerNo;
```

```

} sJob;

int main()
{
    clrscr();
    printf("\nSize of union = %d bytes", sizeof(uJob));
    printf("\nSize of structure = %d bytes", sizeof(sJob));
    getch();
    return 0;
}

```

OUTPUT:

```

Size of union = 32 bytes
Size of structure = 38 bytes

```

RESULT:

Thus the program was successfully executed and verified

<p>Ex. No. 15 Date :</p>	<p>SWAPPING VARIABLE USING POINTER</p>
------------------------------	--

AIM: To write a C program to swap two number using pointer.

ALGORITHM:

```

Step 1: Start the process.
Step 2: Include the appropriate header files.
Step 3: Declare the variable a, b and temp.
Step 4: Declare the pointer variable *ptr1, *ptr2.
Step 5: Get the value of a and b.
Step 6: Print the value a and b.
Step 7: Assign the address of &a to ptr1 and &b to ptr2.
Step 8: Swap the variable
        temp=*ptr1;*ptr1=*ptr2;*ptr2=temp;
Step 9: Print the value a and b.
Step 10: Terminate the process.

```

PROGRAM:

```

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

int main()
{
    int a, b, temp;
    int *ptr1, *ptr2;
    clrscr();
    printf("\nEnter the value of a and b: ");
    scanf("%d %d", &a, &b);

```

```
printf("\nBefore swapping a = %d and b = %d", a, b);

// Assign the memory address of a and b to *ptr1 and *ptr2
ptr1 = &a;
ptr2 = &b;

// Swap the values a and b
temp = *ptr1;
*ptr1 = *ptr2;
*ptr2 = temp;

printf("\nAfter swapping a = %d and b = %d", a, b);
getch();
return 0;

}
```

OUTPUT:

Enter the value of a and b: 10 20

Before swapping a = 10 and b = 20

After swapping a = 20 and b = 10

RESULT:

Thus the program was successfully executed and verified

Prepared by
Dr. A. Subramani
Assistant Professor
Department of Computer Science
M.V. Muthiah Government Arts College for Women
Dindigul