**ACKNOWLEGDEMENT**

I would like to express my special thanks for gratitude to my teacher Mr. Anil

Thakur as well as my principle Mr. Ram Hari Silwal who gave me this golden opportunity

to dothis wonderful project on the topic of C programming and java script, which help me

to learn a lot in the research and course of completion of this project

I am also indebted to my family and friend ( Chandra magar , Aashish Chaulagain , Saroj

Dungana ,Sachin Moktan, Arun Thami , Keshab Bohora , Bal Bahadur Jaisi , Prajwal Karki,

Rohit Rasaili) for their valuable support , advice and love which help me to do this project

with the given time frame.

**C Language Introduction**

C is a procedure programming language initially developed by DenisRitchie in the year 1972

at Bell laboratories of AT and T labs. It wasmainly developed as a system programming

language to write theUNIX. C is a powerful general purpose programming language .

C is amiddle level language . C language is used as most common platformfor developing

system programs. It can be used to develop softwarelike : Operating system, Data bases,

Compliers and so on.

There are following characteristics of C languages:

I. C is a structure programming language.

II. C is a general purpose programming language.

III. C has a large and powerful set of the operators.

IV. C has very less number of reserve words.

V. C has ability to extend itself by adding more function to its

library.

**Structure of C program**

#include &lt;stdio.h&gt;

Int main()

{

Int a=5;

printf(“interger number =%d”, a);

Return 0;

}

**Data types in C**

Data types are the means to identify the type of data and associated operations of handling it.

C support different type of data each of which may be represented differently with in the

Computer’s memory. C data types are of 2 types:

I**. Primary Data**

II**. Secondary Data**

**Primary Data**

1. **Character:**

We use the keyword char for character data type. It is used tostore single bit character.

We can store alphabets from A-Z and

0-9 digits using char.

2**. Integer**:

We use the keyword int for integer data type. The int datatypes is used to store non

fractional number which includes positives, negatives and zero values. It means real

number or sample number.

3. **Floating**:

We use the keyword float for floating point data type. Float is used to store decimal

number. It can store up to 6 digits of precision that after decimal point is 6 digit.

4. **String**:

We use the keyword string for string point data type. String is used to store words,

sentence .For example: name, hello, how are you.

**Secondary Data**

i. **ARRAY**

Array is a derived data type which is used to store the data of the same type in a contiguous memory.

We can store any type of datatype ranging from int, string, and structure. That is we can make any

array of primitive, user defined or of derived data type.

ii. **UNIO**:

Union is quite similar to structure as it is also used to store values of multiple data type. the only

difference between structure and union is that in the space is equal to the highest memory required

by the data types.

iii. **Structure**:

Structure is a user define data type in C, we can store values of multiple data types.

**FUNCTION in C Program**

Certainly! In C programming, a function is a block of code that performs a specific task. It is defined with a name, a return type (if applicable), a set of parameters (if any), and a body containing the code to be executed. Here's the basic syntax for defining a function in a C program:

return\_type function\_name(parameter1\_type parameter1\_name, parameter2\_type parameter2\_name, ...) {

// Function body (code to be executed)

// ...

// Return statement (if applicable)

// return some\_value;

}

(1)WAP to find sum of two number

#included<stdio.h>

#include<conio.h>

void sum (int,int);

void main()

{

clrscr();

int a,b;

printf(“enter any two number”);

scanf(“%d,%d”,&a,&b);

sum(a,b);

getch();

}

void sum(int a,int b)

{

int s;

s=a+b;

printf(“ the sum is %d”,sum);

}

(2)WAP to find divide of two number

#included<stdio.h>

#include<conio.h>

void div(int,int);

void main()

{

clrscr();

int a,b;

printf(“enter any two number”);

scanf(“%d,%d”,&a,&b);

div(a,b);

getch();

}

void div(int a,int b)

{

float s;

s=a/b;

printf(“ the sum is %f”,div);

}

(3)WAP to find area of the circle

#included<stdio.h>

#include<conio.h>

void area(int);

void main()

{

clrscr();

int a;

printf(“enter radius of the circle”);

scanf(“%d”,&a);

area(a);

getch();

}

void area(int a)

{

float area;

area=a\*a\*3.14;

printf(“ the area is %f”,area);

}

(4)WAP to find volume of sphere

#included<stdio.h>

#include<conio.h>

void vol(int,int);

void main()

{

clrscr();

int r,h;

printf(“enter any two number”);

scanf(“%d,%d”,&a,&b);

vol(r,h);

getch();

}

void vol(int)

{

Float vol;

vol=3.14\*r\*r\*h;

printf(“ the volume is %d”,vol);

}

(5)WAP to find the number positive or negative

#included<stdio.h>

#include<conio.h>

void num(int);

void main()

{

clrscr();

int a;

printf(“enter any number”);

scanf(“%d”,&a);

num(a);

getch();

}

void num(int a)

{

If (a>0)

printf(“the number%d is positive “,a);

else

printf(“the number is negative”,n);

}

(6)WAP to find the number even,odd,zero

#included<stdio.h>

#include<conio.h>

void num(int);

void main()

{

clrscr();

int a;

printf(“enter any number”);

scanf(“%d”,&a);

num(a);

getch();

}

void num(int a)

{

If(a%2==0)

printf(“%d number is even”,a);

else(a=0)

printf(“%d is zero”,a)”

elseif

printf(“%d is odd”,a):

}

(7)WAP to print first 10 number

#included<stdio.h>

#include<conio.h>

void int();

void main()

{

clrscr();

num();

getch();

}

void int()

{

int I;

for(i=0;i<10;i++)

{

printf(“ %d/n”,i);

}

}

(8)WAP to print first ten even number

#included<stdio.h>

#include<conio.h>

num();

void main()

{

clrscr();

num();

getch();

}

void num()

{

int i,a=2;

for(i=0;i<10;i++);

{

Printf(“%d/n”,i);

a=a+2;

}

}

(9)WAP to find factorial of a number using function

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

void fact (int);

void main()

{

clrscr();

int n;

printf(“enter any number which is greater than zero/n”);

scanf(“%d”,&n);

fact(n);

getch();

}

void fact (int n)

{

int I;

if (n==0)

printf(“factorial is 0/n”);

else

{

for(i=n;i>1;i--)

}

printf(“the factorial is %d”,n);

}

**STRUCTURE**

In C programming, a structure is a user-defined data type that allows you to group together variables of different data types under a single name. It is a way to organize and represent a collection of related data items. The basic syntax for defining a structure in C is as follows:

struct structure\_name {

// Member variables or fields

data\_type1 member1;

data\_type2 member2;

// ... (additional members)

};

some example of structure

(1)WAP to input roll no and name of 10 students and display in proper format

#include<stdio.h>

#include<conio.h>

struct student

{

int roll;

char name[20];

};

void main()

{

structure student s[10];

int I;

for (i=0;i<10;i++)

{

printf(“enter roll no /n”);

scanf(“%d”,&s[i]roll);

printf(“enter name /n”);

scanf(“%s”,&s[i]name);

}

for (i=0;i<10;i++)

{

printf(“roll no is %d” s[i]roll);

printf(“name is %s”, s[i]name);

getch();

}

(2)WAP to input id, name, address, salary and contact of 20 employ and display in proper format

#include<stdio.h>

#include<conio.h>

struct employ

{

int id;

char name[20];

char address[20];

char salary[20];

char contact[20];

};

void main()

{

structure employ s[10];

int I;

for (i=0;i<20;i++)

{

printf(“enter id no /n”);

scanf(“%d”,&s[i]id);

printf(“enter name /n”);

scanf(“%s”,&s[i]name);

printf(“enter address/n”);

scanf(“%s”,&s[i]address);

printf(“enter contact/n”);

scanf(“%s”,&s[i]contact);

printf(“enter salary/n”);

scanf(“%s”,&s[i]salary);

}

for (i=0;i<20;i++)

{

printf(“id no is %d” s[i]roll);

printf(“name is %s”, s[i]name);

printf(“address is %s”, s[i] address);

printf(“contact is %s”, s[i] contact);

printf(“salary is %s”, s[i] salary);

getch();

}

(3)WAP to input id, name, address, salary and contact of 10 employ and final average of salary in proper format

#include<stdio.h>

#include<conio.h>

struct employ

{

char name[20];

char address[20];

char salary[20];

char contact[20];

int id;

};

void main()

{

structure employ s[10];

int I;

for (i=0;i<10;i++)

{

printf(“enter id no /n”);

scanf(“%d”,&s[i]id);

printf(“enter name /n”);

scanf(“%s”,&s[i]name);

printf(“enter address/n”);

scanf(“%s”,&s[i]address);

printf(“enter contact/n”);

scanf(“%s”,&s[i]contact);

printf(“enter salary/n”);

scanf(“%s”,&s[i]salary);

}

for (i=0;i<10;i++)

{

printf(“id no is %d” s[i]id);

printf(“name is %s”, s[i]name);

printf(“address is %s”, s[i] address);

printf(“contact is %s”, s[i] contact);

printf(“salary is %s”, s[i] salary);

}

int total salary=0;

float avg;

for(i=0;i<10;i++)

{

total salary=total salary + s[i] salary;

}

avg salary=total salary/10;

printf(“the avg salary is %f”,avg);

getch();

}

(4) WAP to input id, name, address, salary of 10 employ and find whose salary is more than 50,000 also display in proper format

#include<stdio.h>

#include<conio.h>

struct employ

{

int salary;

char name[20];

char address[20];

char salary[20];

char contact[20];

int id;

};

void main()

{

structure employ s[10];

int I;

for (i=0;i<10;i++)

{

printf(“enter id no /n”);

scanf(“%d”,&s[i]id);

printf(“enter name /n”);

scanf(“%s”,&s[i]name);

printf(“enter address/n”);

scanf(“%s”,&s[i]address);

printf(“enter contact/n”);

scanf(“%s”,&s[i]contact);

printf(“enter salary/n”);

scanf(“%s”,&s[i]salary);

}

for(i=0;i<10;i++)

{

if(s[i] salary>50,000)

printf(“salary is more than 50,000”,s[i]salary);

else

printf(“salary is less than 50,000”, s[i]salary);

}

for (i=0;i<10;i++)

{

printf(“id no is %d” s[i]id);

printf(“name is %s”, s[i]name);

printf(“address is %s”, s[i] address);

printf(“contact is %s”, s[i] contact);

printf(“salary is %s”, s[i] salary);

}

getch();

}

(5)WAP to input roll no, name ,address and masks in 3 subject of 10 students and find who is pass or fail

#include<stdio.h>

#include<conio.h>

struct student

{

int roll no;

char name[20];

char address[20];

float phy , che , math;

}

void main()

{

int I;

struct student s[10];

for(i=0;i<10;i++)

{

printf(“enter roll no”);

scanf(“%d”,&s[i]roll no);

printf(“enter contact no”);

scanf(“%s”,&s[i]contact no);

printf(“enter name”);

scanf(“%s”,&s[i]name);

printf(“enter marks in phy ”);

scanf(“%f”,&s[i]phy);

printf(“enter marks in che ”);

scanf(“%f”,&s[i]che);

printf(“enter marks in math ”);

scanf(“%f”,&s[i]math);

}

for(i=0;i<10;i++)

{

if(s[i]phy>40&&s[i]che>40&& s[i]math)

printf(“%s is pass”, s[i]name);

else

printf(“%s is fail, s[i]name);

}

for(i=0;i<10;i++)

{

printf(“roll no is %d”, s[i] roll no);

printf(“name is %s”, s[i] name);

printf(“address is %s”, s[i] address);

printf(“%f is mask of physic”, s[i]phy);

printf(“%f is mask of chemistry”, s[i]che);

printf(“%f is mask of math”, s[i]math);

}

getch();

}

**FILE in C Program**

In C programming, file handling is an essential aspect that allows you to interact with external files on your computer. Files are used for both input and output operations, and the **stdio.h** library provides functions and structures for efficient file handling. Here's a brief overview of file handling in C:

1. **FILE Structure:**
   * The **FILE** structure represents a file stream in C.
   * It contains information about the file, such as its position, status, and buffering details.
   * Declared as **FILE \*filePointer;**
2. **Opening a File:**
   * The **fopen** function is used to open a file before performing any read or write operations.
   * Syntax: **FILE \*fopen(const char \*filename, const char \*mode);**
   * Example: **FILE \*file = fopen("example.txt", "r");**
3. **Closing a File:**
   * The **fclose** function is used to close an open file stream.
   * Syntax: **int fclose(FILE \*stream);**
   * Example: **fclose(file);**
4. **Writing to a File:**
   * The **fprintf** function is commonly used to write formatted data to a file.
   * Example: **fprintf(file, "Hello, this is a line in the file.\n");**
5. **Reading from a File:**
   * The **fscanf** function is often used for reading formatted data from a file.
   * Example: **fscanf(file, "%s %d", name, &age);**
6. **Error Handling:**
   * Always check if a file is successfully opened before performing operations on it.
7. **Modes:**
   * File modes determine the type of operations that can be performed on the file.
   * Common modes include **"r"** (read), **"w"** (write), and **"a"** (append).
8. **Sequential Access:**
   * File operations in C are typically sequential, meaning you read or write data sequentially from the beginning to the end of the file.
9. **Binary vs. Text Files:**
   * Files can be opened in binary or text mode. Text mode handles newline characters differently between Windows and Unix-like systems, while binary mode treats them uniformly.

File handling in C provides a powerful mechanism for input and output operations, making it possible to integrate external data into your programs and store results for later use. Always remember to close files after usage to release system resources.

The basic syntax for defining a file in C is as follows:

#include <stdio.h>

int main() {

// Declare a FILE pointer

FILE \*filePointer;

// Open a file for reading or writing

// "filename" is the name of the file, and "mode" is the access mode ("r" for read, "w" for write)

filePointer = fopen("filename.txt", "r");

// Check if the file is successfully opened

if (filePointer == NULL) {

fprintf(stderr, "Error opening the file.\n");

return 1; // Return an error code

}

// File operations (reading or writing) go here

// Close the file when done

fclose(filePointer);

return 0;

}Top of Form

(1)WAP to store your name ,id and address in a file “info.txt”

#include<stdio.h>

#include<conio.h>

void main ()

{

clrscr();

FILE\*p;

int id;

char name[20];

char address[20];

p=fopen(“info.txt”);

printf(“enter id no”);

scanf(“%d”,& id);

printf(“enter address”);

scanf(“%s”,& address);

printf(“enter name ”);

scanf(“%s”,& name);

fprintf(“p,%d,%s,%s”, id, address, name);

fclosef(p);

getch();

}

(2)WAP to store name ,id and salary of employees in a file “info.txt”

#include<stdio.h>

#include<conio.h>

void main ()

{

clrscr();

FILE\*p;

int id;

char name[20];

char salary[20];

p=fopen(“info.txt”);

printf(“enter id no”);

scanf(“%d”,& id);

printf(“enter salary”);

scanf(“%s”,& salary);

printf(“enter name ”);

scanf(“%s”,& name);

fprintf(“p,%d,%s,%s”, id, salary, name);

fclosef(p);

getch();

}

(3)WAP to store id, name, address, salary and contact of 10 employee in a file “emp.txt”

#include<stdio.h>

#include<conio.h>

struct employee

{

char name[20];

char address[20];

char salary[20];

char contact[20];

int id;

};

void main()

{

clrscr();

structure employ s[10];

FILE\*p;

int I;

for (i=0;i<10;i++)

{

printf(“enter id no /n”);

scanf(“%d”,&s[i]id);

printf(“enter name /n”);

scanf(“%s”,&s[i]name);

printf(“enter address/n”);

scanf(“%s”,&s[i]address);

printf(“enter contact/n”);

scanf(“%s”,& s[i]address);

printf(“enter salary/n”);

scanf(“%s”,&s[i]salary);

}

for (i=0;i<10;i++)

{

fprintf(“p,%d,%s,%s,%s%s”, s[i]id, s[i]name, s[i]address, s[i]salary);

}

fclose(p);

getch();

}

(4)WAP to store id, name and salary of some employee in a file “emp.txt”

#include<stdio.h>

#include<conio.h>

include<string.h>

void main()

{

char name[20];

int salary;

int id;

char next[20];

FILE\*p;

p=fopen(“emp.txt”,”w+”);

if(p==null)

{

printf (“file doesnot exit\n”);

exit;

}

do

{

printf(“enter id no”);

scanf(“%d”,& id);

printf(“enter salary”);

scanf(“%s”,& salary);

printf(“enter name ”);

scanf(“%s”,& name);

fprintf(“p,%d,%s,%s”, id, salary, name);

printf(“type yes if you want to add more \n“)

scanf(“%s”,& next);

while (strcmp(next:”yes”)=0);

rewind(p);

while (fscanf(“p,%d,%s,%s”,& id,& salary, &name=null);

{

printf(“id=%d\t, salary=%s\t, name=%s\t”, id ,salary name);

}

fclose(p);

getch();

}

**JAVA SCRIPT**

JavaScript is a versatile programming language primarily used for web development to add interactivity and dynamic behavior to web pages. Here's a short note on JavaScript:

1. **Client-Side Scripting:**
   * JavaScript is primarily employed as a client-side scripting language, allowing developers to write code that runs in a user's browser. This enables dynamic updates and interactions without the need for server-side processing.
2. **Syntax:**
   * JavaScript syntax is similar to other C-based languages like Java and C++. It's designed to be easy to learn for those familiar with programming.
3. **Variables and Data Types:**
   * Variables are used to store data, and JavaScript is loosely typed, meaning variables can hold values of any data type without explicit type declarations.
4. **Functions:**
   * Functions in JavaScript allow for code organization and reusability. They can be defined using the **function** keyword and invoked as needed.
5. **Event-Driven Programming:**
   * JavaScript is often used for handling events, such as user actions on a webpage. Event listeners can be attached to respond to user interactions like clicks, mouse movements, and keyboard input.
6. **DOM Manipulation:**
   * The Document Object Model (DOM) represents the structure of an HTML document. JavaScript can dynamically manipulate the DOM, allowing developers to change content, structure, and style of a webpage in response to user actions.
7. **Asynchronous Programming:**
   * JavaScript supports asynchronous programming using callbacks, promises, and more recently, async/await. This is crucial for handling tasks like fetching data from servers without blocking the user interface.
8. **Libraries and Frameworks:**
   * Numerous libraries and frameworks built on JavaScript, such as jQuery, React, Angular, and Vue.js, simplify web development tasks and enable the creation of complex, interactive web applications.
9. **JSON (JavaScript Object Notation):**
   * JavaScript uses JSON for data interchange. JSON is a lightweight data format that is easy for humans to read and write, and easy for machines to parse and generate.
10. **Cross-Browser Compatibility:**
    * JavaScript is designed to run on various browsers, ensuring cross-browser compatibility. However, developers need to be mindful of potential differences in browser implementations.
11. **Server-Side Development:**
    * With the advent of technologies like Node.js, JavaScript is now used for server-side development as well. It allows developers to use the same language on both the client and server sides.

JavaScript's ubiquity in web development and its continuous evolution make it a crucial language for building interactive and responsive web applications.

Top of Form

(1)write a java script program to find the area of circle

<html>

<head>

<title>area</title>

</head>

<body>

<script>

var r= praseInt(prompt("enter radius"));

var x=3.14;

document.write("area of circle is ",r\*r\*x);

</script>

</body>

</html>

(2) write a java script program to find volume of rectangle

<html>

<head>

<title>volume</title>

</head>

<body>

<script>

function volume(l,b,h)

{

var l=praseInt(prompt("enter length"));

var b=praseInt(prompt("enter breadth"));

var h=praseInt(prompt("enter height"));

document.write("volume of rectangle is ",l\*b\*h);

}

volume();

</script>

</body>

</html>

(3)write a java script program to find the number is positive, negative

<html>

<head>

<title>positive or negative </title>

</head>

<body>

<script>

var n= praseInt(prompt("enter number"));

if(n>0)

document.write("the number is positive ");

else

document.write("the number is negative ");

</script>

</body>

</html>