o Write an essay covering the history and evolution of C programming. Explainits importance and why it is still used today.

- C programming language develop in 1972 by Dennis Ritchie at bell laboratories in usa.
- C is pop(Procedure Oriented Programming) language . procedure oriented programming is function,method and block of code .c language is pop language.
- C is high-level programming language , human understable and easy to read write ,access to the uaser.
- This is a flexible , portability , efficiency operating system , embedded system and application.
- C language is key sensitive language. And extension .c file create.
- C language basic structure :

```
#include<stdio.h>
main()
{
    printf("\n\n\t hello");
}
```

Describe the steps to install a C compiler (e.g., GCC) and set up an IntegratedDevelopment Environment (IDE) like DevC++, VS Code, or CodeBlocks.

1.Google search the c compiler or IDE.

DEV C++,VS CODE,CODEBLOCK etc…

2.choice IDE:

Dev c++ : download and install the dev c++ valid link in browser and click download button.

Vs code : download and install the vs code valid link in browser and click download button.

Codeblocks : download and install the codeblock valid link in browser and click download button.

3.configure in IDE :

Dev c++ : install the dev c++ after the create new file and write code.

Vs code : install the vs code after the c language extension install and create new file write code.

Codeblock : install the codeblock after the create new file and write code.

4.save a program and run :

Dev c++ : first program execute in dev c++. .c extension file save and run program.

Vs code : first program execute in vs code. .c extension file save and run program.

Codeblock : first program execute in codeblock. .c extension file save and run program.

o Explain the basic structure of a C program, including headers, main function, comments, data types, and variables. Provide examples.

Basic structure in c:

```
#include<stdio.h>

main()
```

```
{
        printf("\n\n\t hello");
}
```

Program is set of instruction for the computer perform specific task. #include<stdio.h> is library,

: this is preprocessor , include : keyword in c, <stdio.h> : standard input output header file ,

Main() : function and start the program from here , { } : block of code within write the statementand condition.

Comments : comments is part of program but not display output. Comment is use explain or instruction code.

comment are two type 1. Single line comments 2. Multiline comments

1. Single line comments : single line comments start // use symbol.

2. Multiline comment : multi line comments start /* and end */ use symbol.

Data type: data type is type of data and store variable and value.

Datatype is two type : 1.primitive datatype.

2.nonprimitive datatype.

Primitive datatype : ptimitive datatype is provide ianguage.

Ex : int , float , char etc…

nonprimitive datatype : nonptimitive datatype is provide developer.

Ex : string , array , structure etc…

Integer : declare %d store the positive value small range.

Float : declare %f store the decimal value small range.

Long integer : declare %ld store the positive value high range.

double : declare %lf store the decimal value high range.

charcter: declare %c store the single character value.

String : declare %s store the group of character value.

Variable : element(memory) to store the particule value .

Ex : a=10

a=123456789

a=34.56

a=345678.7656

a='g'

a="tops"

example:

```
//value store
include<stdio.h>
Main()
{
Int a=100;
Float b=200;
Printf("\n store the value: %d",a);
Printf("\n store the value: %f",b);
}
```

o Write notes explaining each type of operator in C: arithmetic, relational, logical, assignment, increment/decrement, bitwise, and conditional operators.

Operator : operator is symbol and perform the operation from value.

Multiple operator in c language

1.arithmatic operator : perform the mathematical operation use arithmetic operator.

Ex : +(addition) , -(substarction) , *(multiplication), /(division) , %(parectege).

2. relational operator: perform the two value comparison use relational operator.

Ex : ==(double equal) , !=(not equal) , >(greter then) , <(less then) , <=(lessthen equal) , =>(greterthen equal)

3.logical operator : perform the two or more condition check use logical operator.

Ex : &&(and) , ||(or) , !(not)

4.assignment operator : perform the assign value use assignment operator.

Ex: += , -+, *= , /=

a+=b = a=a+b

a-=b = a=a-b

a*=b = a=a*b

a/=b = a=a/b

5.increment/decrement :increment operator value+1,decrement operator value-1

Prefix : ++a , --a postfix : a++ , a--

6.bitwise operator : ex : & , | , << , >>

o Explain decision-making statements in C (if, else, nested if-else, switch). Provide examples of each.

1. If stetment : use the if stetment cheak only true condition check .

```
Syntex :  if(condition)
          {
              //stetment
          }
```

2.if_else stetment : use the if_else stetment check the true or false condition.

```
Syntex : if(condition)
         {
             //stetment
         }
           Else
         {
             //stetment
         }
```

3.nested if stetment : use the nested if stetment check if condition within if condition (if into if)

```
Syntex : if(condition)
```

```
{
If(condition)
{
//stetment
}
 else
{
        //stetment
}
else
{
        //stetment
}
```

4.switch stetment : multiple choice value and user one choice use switch stetment.

Not use relation operator , switch use int , charcter datatype , switch use keyword switch , break , case , default.

Syntex : switch(choice)

```
{
Case 1:  // stetment
Break;
Case 2 : //stetment
Break;
Default : //stetment
}
```

o Compare and contrast while loops, for loops, and do-while loops. Explain the scenarios in which each loop is most appropriate.

While loop : while(condition) { execute the code } . while loop condition is true loop is execute and condition is false loop is not execute. while loop is entry control loop, entry control loop is check condition before execute code. Exit the loop condition is false.

For loop : for(initialization , condition , increment/decrement). for loop condition is true loop is execute and condition is false loop is not execute. for loop is entry control loop, entry control loop is check condition before execute code. Exit the loop condition is false.

Do while loop : do { execute the code } while(condition) . do while loop condition first time execute and after condition is true loop is execute and condition is false loop is not execute. do while loop is exit control loop, exit control loop is execute code before check condition. Exit the loop condition is false.

While loop : I = 1;(initialization) : starting point

```
While(I <= 5)condition : ending point
{   printf ("%d", i); //execute code
     I++; : increment/decrement
}
```

For loop : (I = 1(initialization; I <= 5 (condition) ; i++(increment/decrement))

```
{
```

```
        printf ("%d" , i); // execute code
       }
```

```
Do while loop : I = 1;(initialization)
Do{
      printf ("%d" , i); // execute code
      I++; : increment/decrement
    }
     While (I <= 5); : (condition)
```

Explain the use of break, continue, and goto statements in C. Provide examples of each.

Break : break is keyword and break keyword use the break the code , rest of the code will not executed.

```
        //break
#include<stdio.h>
int main()
{
int i;
for(i=0;i<10;i++)
{
        if(i==6)
        break;
```

```
        printf("\n%d",i);
}}
```

continue : continue is keyword and continue keyword use the skip the code , rest of the code will be executed.

```
//continue
#include<stdio.h>
int main()
{
int i;
for(i=0;i<10;i++)
{
        if(i==6)
        continue;
        printf("\n%d",i);
}
}
```

Goto : goto is keyword and goto is label to continue the code execute.

```
#include<stdio.h>
main()
{
int i;
i=0;
abc:
        printf("\n%d",i);
```

```
        i++;
            if(i<=10)
            goto abc;
}
```

o What are functions in C? Explain function declaration, definition, and how to call a function. Provide examples.

Function is block of code in c. use the function () round breaket .

function is two type :

1) Build in function : build in function is already c language provide.it is not created by developer.
2) User define function : user define function is provide by developer.

- **Mainly four type of user define function:**

1) without argument and without return value
2) without argument and with return value
3) with argument and without return type
4) with argument and with return type

- **Function mainly three part :**
  - Function Declaration
  - Function call
  - Function definition

Example:

```
#include<stdio.h>
Void main()
Main()
{
```

```
                Num();
}
Void num()
{
        Int I;
        For(i=0;i<10;i++)
{
        Printf("%d",i);
}
```

o Explain the concept of arrays in C. Differentiate between one-dimensional and multi-dimensional arrays with examples.

Array : array is collaction of element a different data type.

Array is three type :

1) one dimensional array
2) two dimension array
3) multi dimension array

Differentiate between one-dimensional and multi-dimensional arrays:

One dimensional : it has only one dimensional. Array store single line multiple element. One dimensional array is execute series type.

Multi dimensional : it has three or multiple dimensional. Multi dimensional array store row and column multiple element. And multi dimensional array is execute table or matrix.

Example:

```
int array[50],i,n;
printf("enter user value");
scanf("%d",&n);

for(i=0;i<n;i++)
    {
        printf("enter the value: [%d]",i);
        scanf("\n%d",&array[i]);
    }

    for(i=0;i<n;i++)
    {
        printf("\n array[%d] %d",i,array[i]);
    }
```

o Explain what pointers are in C and how they are declared and initialized. Why are pointers important in C?

pointer : pointer is variable store the memory address to another variable.

Pointer is used to point the actual value.

declared and initialized : datatype *pointer variable

```
int a=10;
int *p=&a;
printf("\n\n\t a = %d", a);
printf("\n\n\t p = %d", *p);
printf("\n\n\t p = %p", p);
```

Why are pointers important in C:

o Explain string handling functions like strlen(), strcpy(), strcat(), strcmp(), and strchr(). Provide examples of when these functions are useful.

strlen() : find out the length of the string

Example :

```
char str[20];
printf(“enter the string”);
scanf(“%d”,&str”);
printf("\n\n\t Length of String : %d", strlen(str));
```

strcpy() : to copy one string to another.

Example :

```
char str1[20], str2[20];
printf(“enter the string1”);
scanf(“%d”,&str1”);
printf("\n\n\t String2 : %s", strcpy(str2, str1));
```

strcat() : To concate two strings.

Example :

```
char str1[20], str2[20], str3[20];
printf(“enter the string1”);
scanf(“%d”,&str1”);
```

```
printf(“enter the string2”);
scanf(“%d”,&str2”);
strcpy(str3, strcat(str1, str2));
printf("\n\n\t String3 : %s", str3);
```

strcmp() : To compare two strings. (by default refer the case).

Example :

```
Int str1[20],str2[20];
if(strcmp(str1, str2)==0)
        printf("\n\n\t Strings are equal.");
else
        printf("\n\n\t Strings are not equal.");
```

strchr() : return a pointer position of the first charchter in string.

Example :

```
char str[]="hello world";
char *ptr=strchr(str,'w');
if(ptr!=NULL)
{
        printf("%s",ptr);
}
```

o Explain the concept of structures in C. Describe how to declare, initialize, and access structure members.

Structures : Structure is a template or blueprint, and structure is collection of elements with different data type.

Declare : struct structname

```
{
    Datatype membername;
    Datatype membername;
}
```

Initialize:

```
struct student
{
    int rollno;
    char name[20];
    char gread;
};
```

access structure members : s.rollno , s.name , s.gread access structure member.

o Explain the importance of file handling in C. Discuss how to perform file operations like opening, closing, reading, and writing files.

file handling : file handling in c process the create , open , read , write , close operation in file .provide different function fopen() , fread() , fwrite() etc.

read : write :

```
FILE *fptr;
fptr=fopen("File1.txt","r");
```

```
FILE *fptr;
fptr=fopen("File1.txt","w");
```