Question answer of c-programming:

2068 Baishak

1. ***What is high level languages? Writ the different types of high level languages with their examples. How is compiler different than programming languages?***

Answer:

High-level languages are a type of languages are a type of programming language that is designed to be more human-readable and easier to use than low-level languages like assembly or machine code. They provide abstractions and higher-level constructs that make it simpler for programmers to express their ideas and create complex software without needing to deal with the low-level details of the computer's architecture.guage that is designed to be more human-readable and easier to use than low-level languages like assembly or machine code. They provide abstractions and higher-level constructs that make it simpler for programmers to express their ideas and create complex software without needing to deal with the low-level details of the computer's architecture.

Here are some different types of high-level programming languages with examples:

Python: Known for its simplicity and readability, Python is widely used in various fields, including web development, scientific computing, data analysis, and more.

Java: A popular language used in many areas, particularly in building large-scale applications, mobile apps (Android), and web applications.

C++: An extension of the C language, C++ is commonly used in system programming, game development, and other performance-critical applications.

JavaScript: Primarily used for web development to add interactivity to websites, JavaScript is a key component of modern web applications.

Ruby: Known for its elegant syntax and productivity, Ruby is often used in web development, and the Ruby on Rails framework is widely used for building web applications.

C#: Developed by Microsoft, C# is used in various applications, including Windows desktop applications and game development using the Unity engine.

Swift: Developed by Apple, Swift is used for building iOS, macOS, watchOS, and tvOS applications.

PHP: A server-side scripting language primarily used for web development to create dynamic web pages.

A compiler is a type of software that translates the source code written in a high-level programming language into machine code or another lower-level representation that the computer can execute. In contrast, a programming language is a formal language with a set of rules and syntax used for writing software programs. The compiler is a tool that takes the human-readable code written in a programming language and converts it into executable code that the computer can run. Essentially, the compiler bridges the gap between the high-level programming language and the machine code executed by the computer's hardware.

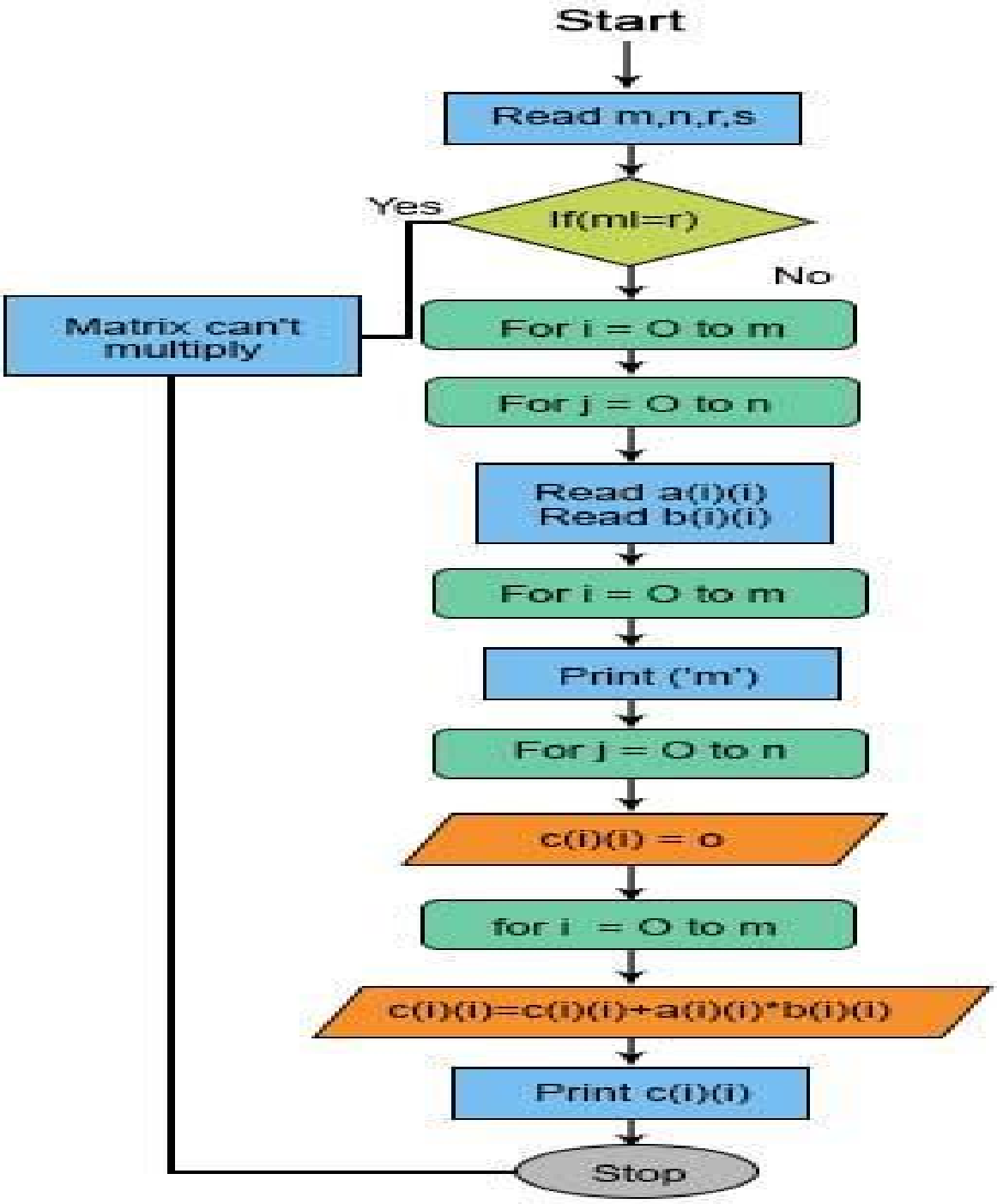
2. ***What is pseudo code? How it is different from algorithm? Draw the flow chart to slove multiplication if two matrices?***

Answer:

Pseudocode is a high-level, human-readable representation of an algorithm or a step-by-step procedure for solving a problem. It is not a formal programming language but rather a simplified description of the logic and steps involved in solving a problem. Pseudocode helps programmers outline their approach before translating it into actual code in a specific programming language.

Algorithm is a broader concept that refers to a well-defined, finite set of instructions or rules that define a sequence of operations to solve a problem or perform a task. Algorithms can be represented in various ways, including pseudocode, flowcharts, or actual code in a programming language. Pseudocode is one way to describe an algorithm, but it is not the only way.

A flowchart is a visual representation of a process or algorithm that uses different shapes and arrows to depict the steps and their relationships. It's a way to show the flow of control and decision points in a structured manner. Let's create a simple flowchart for matrix multiplication:


Start
Read m,n,r,s
Yes
If(ml=r)
No
Matrix can't multiply
For i = O to m
For j = O to n
Read a(i)(i)
Read b(i)(i)
For i = O to m
Print ('m')
For j = O to n
c(i)(i) = o
for i = O to m
c(i)(i)=c(i)(i)+a(i)(i)*b(i)(i)
Print c(i)(i)
Stop

3. ***Write the hierarchy of operation of the following expression in computer programming C: !EOF ||B.Salary + Daily_Allowances > 8000 && eligible_code == 1|| net_pay>10000. Write differential formatted input output used in For Tran programming language.***

Answer:

```
program InputOutputExample
  implicit none
  character(10) :: eligible_code
  real :: B_Salary, Daily_Allowances, net_pay

  ! Input
  write(*, *) "Enter B.Salary:"
  read(*, *) B_Salary

  write(*, *) "Enter Daily_Allowances:"
  read(*, *) Daily_Allowances

  write(*, *) "Enter eligible_code (1 or 0):"
  read(*, *) eligible_code

  write(*, *) "Enter net_pay:"
  read(*, *) net_pay

  ! Output
  if (.not.EOF .or. B_Salary + Daily_Allowances > 8000 .and. eligible_code == 1 .or. net_pay > 10000) then
    write(*, *) "Eligible for something."
  else
    write(*, *) "Not eligible."
  end if
end program InputOutputExample
```

4. ***Write a program in C to calculate the frequency (F) for different values of capacitances(C), for a certain electrical circuit, with an inductance (L) and resistance(R), the damped natural frequency is given BY F= $\sqrt{((1/LC) - (R^2/4C^2))}$. It is required to study th variation of the frequency with capitance starting from 0.01 to0.1 in steps and calculation should be done using defined function.***

Answer:

```
#include <stdio.h>

#include <math.h>


// Function to calculate the frequency (F) using the given formula

double calculateFrequency(double L, double R, double C) {

   return sqrt(1 / (L * C) - (pow(R, 2) / (4 * pow(C, 2))));

}


int main() {

   double L, R, C;

   double capacitance;


   // Given values for the inductance and resistance

   L = 0.5; // Example value for inductance (you can change this)

   R = 10;  // Example value for resistance (you can change this)


   // Calculate and display the frequency for different capacitance values

   for (capacitance = 0.01; capacitance <= 0.1; capacitance += 0.01) {

      double frequency = calculateFrequency(L, R, capacitance);

      printf("Capacitance (C) = %.2f, Frequency (F) = %.6f\n", capacitance, frequency);

   }


   return 0;

}
```

5. *Write a syntax in c-programming with example of the following:"scanf", single statement "do…while","strepy", structures with arrays.*

Answer:

scanf:

The scanf function is used for reading input from the standard input (usually the keyboard) in C. It allows you to read formatted input, which means you can specify the data type you're expecting to read and the corresponding variable where the value will be stored.

Example of using scanf to read an integer from the user:

```
#include <stdio.h>

int main()

{ int number;

printf("Enter an integer: ");

scanf("%d", &number);

printf("You entered: %d\n", number);

return 0; }
```

Single statement do...while:

The do...while loop is a type of loop in C that ensures the loop body is executed at least once, even if the condition is false after the first iteration.

Example of using a single statement do...while loop to print numbers from 1 to 5:

```
#include <stdio.h>

int main()

{ int i = 1;

do

{

printf("%d ", i); i++;

}

while (i <= 5);

printf("\n");
```

```
 return 0; }
```

Structures with Arrays:

In C, you can have structures that contain arrays as members. This allows you to create more complex data structures that hold multiple types of data.

Example of a structure with an array member:

```
#include <stdio.h>

struct Student { char name[50];

int rollNo;

 float marks[5]; // Array of marks for 5 subjects };

 int main()

{ struct Student student1;

// Input student data

 printf("Enter student name: ");

scanf("%s", student1.name);

 printf("Enter roll number: ");

scanf("%d", &student1.rollNo);

printf("Enter marks for 5 subjects:\n");

for (int i = 0; i < 5; i++)

{

 printf("Subject %d: ", i + 1);

 scanf("%f", &student1.marks[i]);

}

 // Display student data

printf("\nStudent Details:\n");

 printf("Name: %s\n", student1.name);

 printf("Roll Number: %d\n", student1.rollNo);

 printf("Marks:\n");
```

```
for (int i = 0; i < 5; i++)

{

printf("Subject %d: %.2f\n", i + 1, student1.marks[i]);

}

return 0;

}
```

In this example, we define a structure Student that has members for the student's name, roll number, and an array of marks for 5 subjects. We read this data from the user and then display it.

6. *What are the rules that govern the passing of array to users to users defined function? Compare "user defined function" and "switch" in c programming languages.*

Answer:

When passing arrays to user-defined functions in C, it's important to keep in mind that arrays are passed by reference, which means that the function receives a pointer to the original array, not a copy of the entire array. Here are some rules to consider when passing arrays to functions:

Size Information: The size of the array is not passed to the function. You need to either pass the size as a separate parameter or use a sentinel value to indicate the end of the array.

Pointer: The array is passed as a pointer to the first element of the array. This pointer allows the function to access the elements of the array.

Modifications: Any modifications made to the array elements within the function are reflected in the original array since the function works with the same memory locations.

Example of passing an array to a user-defined function:

```
#include <stdio.h>

void printArray(int arr[], int size)

{

  for (int i = 0; i < size; i++) {

printf("%d ", arr[i]);

  }
```

```
    printf("\n");
}
int main() {
int numbers[] = {1, 2, 3, 4, 5};
int size = sizeof(numbers) / sizeof(numbers[0]);
printArray(numbers, size);
    return 0;
}
```

7. What are the importance of pointer in c programming languages? Write the input pf the following program.

```
#include <stdio.h>

void main() {
   int k;
   int a[] = {1, 2, 3};
   int *b[3];
   int **c[3];

   clrscr();

   for (k = 0; k < 3; k++) {
      b[k] = &a[k];  // Assign the address of each element in 'a' to the array 'b'
      c[k] = &b[k];  // Assign the address of each element in 'b' to the array of pointers 'c'
      printf("%3d", *b[k]);  // Print the value pointed to by 'b[k]' (which is a[k])
      printf("%3d", **c[k]); // Print the value pointed to by 'c[k]' (which is *b[k] or a[k])
        printf("%3d\n", ***c[k]); // Print the value pointed to by '***c[k]' (which is **c[k] or *b[k] or
a[k])
   }
}
```

Answer :

Pointers are a fundamental concept in C programming, and they hold significant importance for several reasons:

Dynamic Memory Allocation: Pointers allow you to dynamically allocate memory at runtime, enabling you to create data structures of variable sizes, such as arrays and linked lists.

Passing by Reference: When you pass a pointer to a function, you can modify the original data within the function, which is especially useful when working with large data structures or when you want to avoid making copies.

Efficiency: Pointers allow you to directly access and manipulate memory locations, which can lead to more efficient code, particularly when dealing with arrays or when you need to perform low-level operations.

Advanced Data Structures: Pointers are essential for creating complex data structures like trees, graphs, and dynamic data structures, which are not easily represented with just regular variables.

Interfacing with Hardware: Pointers are often used to interface with hardware, manage memory-mapped devices, or directly access specific memory locations.

String Manipulation: C uses pointers to manipulate strings and handle character arrays efficiently.

Answer of the code is :

1 1 1

2 2 2

3 3 3

8. ***WAP a program to read name and write age of 10 different students as the two memebers of a structure named "students". Display the name and corresponding age of the student in an alphabetical order.***

Answer:

```
#include <stdio.h>

#include <string.h>

// Define the structure for a student

struct Student {

   char name[50];

   int age;

};

// Function to compare two students' names for alphabetical sorting

int compareNames(const void *a, const void *b) {

   return strcmp(((struct Student *)a)->name, ((struct Student *)b)->name);

}
```

```
int main() {
   int i;
   struct Student students[10];
   // Input the names and ages of 10 students
   for (i = 0; i < 10; i++) {
  printf("Enter name of student %d: ", i + 1);
   scanf("%s", students[i].name);
  printf("Enter age of student %d: ", i + 1);
   scanf("%d", &students[i].age);
   }
   // Sort the students array based on names
   qsort(students, 10, sizeof(struct Student), compareNames);
   // Display the sorted names and corresponding ages
   printf("\nStudents in alphabetical order:\n");
   for (i = 0; i < 10; i++) {
   printf("Name: %s, Age: %d\n", students[i].name, students[i].age);
   }
   return 0;
   }
```

9. WAP a program in fortan to test whether the accepted year is leap year or not. (Hint: A year is leap if : the year is divisible by 4, but no bu 100 or the year is divided by 400).

Answer:

```
program LeapYearCheck
   implicit none
   integer :: year
 ! Input the year
   write(*, *) "Enter a year: "
   read(*, *) year
   ! Check if the year is a leap year
```

```
    if (mod(year, 4) == 0 .and. (mod(year, 100) /= 0 .or. mod(year, 400) == 0)) then
        write(*, *) year, " is a leap year."
    else
        write(*, *) year, " is not a leap year."
    end if
end program LeapYearCheck
```