C PROGRAMMING 2022 SOLVE PAST PAPER

write the name of two high priority logical operators

In the C programming language, there are three logical operators:

1. Logical AND (`&&`): This operator returns true if both operands are true, and false otherwise. It has a higher priority than the logical OR operator.

2. Logical OR (`||`): This operator returns true if at least one of the operands is true, and false otherwise.

These two logical operators are often used in conditional statements and expressions to control the flow of the program based on certain conditions.

write name of right associative operator

In C programming, the assignment operator (`=`) is an example of a right-associative operator. It is used to assign a value to a variable.

Right-associative means that when multiple operators of the same precedence appear in an expression, they are evaluated from right to left. For example, in the expression `a = b = c`, the assignment operator `=` is right-associative. The value of `c` would be assigned to `b`, and then the value of `b` would be assigned to `a`.

evaluate the expression 15 + 7 - 8 * 4 / 9 % 3 - 2

To evaluate the given expression: 15 + 7 - 8 * 4 / 9 % 3 - 2, we need to follow the order of operations, also known as operator precedence. The order of precedence is as follows:
1. Parentheses ( )
2. Multiplication (*), Division (/), and Modulus (%)
3. Addition (+) and Subtraction (-)

Let's break down the expression step by step:

Step 1: 8 * 4 = 32
Step 2: 32 / 9 = 3 (integer division)
Step 3: 3 % 3 = 0 (remainder after division)
Step 4: 15 + 7 = 22
Step 5: 22 - 0 = 22
Step 6: 22 - 2 = 20
Therefore, the value of the expression 15 + 7 - 8 * 4 / 9 % 3 - 2 is 20.

write a statement in C that shows explicit type casting

float a = 3.14;
int b = (int) a;

int * ptr , var =20; is the statement ptr = var; is true or false

The statement `ptr = var;` is false in this context.
In the given code snippet `int *ptr, var = 20;`, `ptr` is a pointer to an integer (`int *`), and `var` is an integer.
To assign the address of `var` to the pointer `ptr`, you would use the address-of operator (`&`):
ptr = &var;

what is the scop and life time of static variable

Scope: A static variable declared within a function or block has a scope limited to that function or block. It is not accessible outside of the function or block where it is declared.
Lifetime: A static variable has a longer lifetime compared to automatic (local) variables. It retains its value between different function calls and is initialized only once.

easy:
A static variable in C has a limited scope within the block or function where it is declared. It retains its value between different invocations of that block or function. The static variable exists for the entire duration of the program, maintaining its value throughout.

int * ptr ; float var =20; is the statement ptr = &var; is true or not give reason

The statement `ptr = &var;` is **false** in this context.

In the given code snippet `int *ptr; float var = 20;`, `ptr` is a pointer to an integer (`int *`), and `var` is a float.

The reason why the statement `ptr = &var;` is false is that you cannot assign the address of a float variable to a pointer to an integer. Pointers are type-sensitive, and the types of the pointer and the variable it points to must match.

what is output: int p=1; for (int =6 ; i>= 2 ; i = i-2) p*=i; printf("%d\n",p);

48

- Iteration 1: `p = p * i` -> `p = 1 * 6` -> `p = 6`
- Iteration 2: `p = p * i` -> `p = 6 * 4` -> `p = 24`
- Iteration 3: `p = p * i` -> `p = 24 * 2` -> `p = 48`

LONG QUESTIONS

write a program that accept 10 integer values from user in an array and passes the array and its size to a function. The function counts the number even values within array and returns to main() where the result is displayed

```
#include <stdio.h>

int countEvenValues(int arr[], int size) {
    int count = 0;
    for (int i = 0; i < size; i++) {
        if (arr[i] % 2 == 0) {
            count++;
        }
    }
    return count;
}

int main() {
    int arr[10];
    printf("Enter 10 integer values:\n");

    for (int i = 0; i < 10; i++) {
        scanf("%d", &arr[i]);
    }

    int evenCount = countEvenValues(arr, 10);
    printf("Number of even values: %d\n", evenCount);

    return 0;
}
```

write a program that declares a structure employee with data members name (array of characters) and salary (float). create three varaiables of employee , accept values from keyboard and display name of the employess having salary more thean average salary of given employees

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

#define MAX_EMPLOYEES 3

struct Employee {
    char name[50];
    float salary;
};

int main() {
    struct Employee employees[MAX_EMPLOYEES];
    float totalSalary = 0.0;
    float averageSalary;
    int count = 0;

    // Accepting input for each employee
    for (int i = 0; i < MAX_EMPLOYEES; i++) {
        printf("Enter details for Employee %d:\n", i + 1);
        printf("Name: ");
       scanf("%s", employees[i].name);

        printf("Salary: ");
        scanf("%f", &employees[i].salary);
        getchar();  // Clear newline character from input buffer
        totalSalary += employees[i].salary;
        count++;
    }

    // Calculating average salary
    averageSalary = totalSalary / count;

    // Displaying names of employees with salary greater than average
    printf("\nEmployees with salary greater than average salary:\n");
    for (int i = 0; i < MAX_EMPLOYEES; i++) {
        if (employees[i].salary > averageSalary) {
            printf("%s\n", employees[i].name);
        }
    }
  return 0;
}
```

write a program that accepts a +ve integer value in main function and passes the number to a function that calculates the factorial of the number . the main() Displays the factorial . (use while loop to calculate the factorial)

```
#include <stdio.h>

int  calculateFactorial(int num) {
    int  factorial = 1;
    int i = 1;

    while (i <= num) {
        factorial *= i;
        i++;
    }
    return factorial;
}
int main() {
    int number;
    printf("Enter a positive integer: ");
    scanf("%d", &number);
    if (number < 0) {
        printf("Invalid input! The number must be positive.\n");
        return 1;
    }
    int result = calculateFactorial(number);
    printf("Factorial of %d is: %d\n", number, result);
    return 0;
}
```

Here's a program that accepts two float values from the user, passes their pointers to a function, swaps their values within the function, and then displays the swapped values in the `main()` function:

```
#include <stdio.h>

void swapFloats(float *ptr1, float *ptr2) {
    float temp = *ptr1;
    *ptr1 = *ptr2;
    *ptr2 = temp;
}

int main() {
    float num1, num2;

    printf("Enter the first float value: ");
    scanf("%f", &num1);

    printf("Enter the second float value: ");
    scanf("%f", &num2);

    printf("Before swapping: num1 = %.2f, num2 = %.2f\n", num1, num2);

    swapFloats(&num1, &num2);

    printf("After swapping: num1 = %.2f, num2 = %.2f\n", num1, num2);

    return 0;
}
```

write a program that accepts a string from user and pass it to a user defined function named counter(). the function counts the occurrence of character 'M' within the string . the resukt is displayed by the main().

```
#include <stdio.h>

int counter(const char* str) {
    int count = 0;
    for (int i = 0; str[i] != '\0'; i++) {
        if (str[i] == 'M') {
            count++;
        }
    }
    return count;
}
int main() {
    char str[100];

    printf("Enter a string: ");
     scanf("%99s", str);

    int result = counter(str);
    printf("Occurrences of 'M' in the string: %d\n", result);
    return 0;
}
```