Data types

1)Write a program in python that declares an int, a float, and a char variable, initializes them with values, and prints them to the console.

Algorithm:

1. Start
2. Declare an integer variable and assign a value
3. Declare a float variable and assign a value
4. Declare a character variable and assign a value
5. Print the values of all variables
6. End

Flowchart:

```
Start
  ↓
Declare int, float, and char variables
  ↓
Assign values to the variables
  ↓
Print values of int, float, and char
  ↓
End
```

Python Program:

```
# Declaring variables

integer Var = 10      # Integer variable

float Var = 25.5      # Float variable

char_var = 'A'        # Character variable


# Printing the values

print ("Integer Value:", integer_var)

print ("Float Value:", float_var)

print ("Character Value:", char_var)
```

2)Create a program that reads an integer from the user, converts it to a float, and prints both the original integer and the converted float.

Algorithm:

1. Start
2. Prompt the user to enter an integer
3. Read the integer input from the user
4. Convert the integer to a float
5. Print both the original integer and the converted float
6. End

Python Program:

```
# Read an integer from the user

integer value = int (input("Enter an integer: "))


# Convert the integer to a float

float value = float (integer value)


# Print both values

print ("Original Integer:", integer value)

print ("Converted Float:", float value)
```

3) Write a program that uses the size of operator to print the size of various data types like int, float, double, char, short, and long.

Algorithm:

1. Start
2. Import the sys module for size checking
3. Declare variables of different data types (int, float, char, short, long, double)
4. Use sys.getsizeof() to get the size of each variable
5. Print the sizes of all data types
6. End

Python Program:

```
import sys

# Declaring variables
int_var = 10          # Integer
float_var = 10.5      # Float
char_var = 'A'        # Character (stored as a string in Python)
short_var = int_var   # Python does not have a 'short' type, so using int
long_var = 10**10     # Large integer (Python int can handle long values)
double_var = 10.5     # Python's float is equivalent to C's double

# Printing sizes
print("Size of int:", sys.getsizeof(int_var), "bytes")
print("Size of float:", sys.getsizeof(float_var), "bytes")
print("Size of char:", sys.getsizeof(char_var), "bytes")
print("Size of short (int in Python):", sys.getsizeof(short_var), "bytes")
print("Size of long (large int):", sys.getsizeof(long_var), "bytes")
print("Size of double (float in Python):", sys.getsizeof(double_var),
"bytes")
```

Explanation:

- **Python does not have short, long, or double explicitly**, but:
    - short is represented using int.
    - long can be represented using a large integer.
    - double is represented using float in Python.

4)Write a program that takes two unsigned int variables, performs bitwise AND, OR, and XOR operations on them, and prints the results.

Algorithm:

1. Start
2. Declare two unsigned integer variables
3. Perform the following bitwise operations:
    - **AND (&)**
    - **OR (|)**
    - **XOR (^)**
4. Print the results of each operation
5. End

Python Program:

```
# Take two unsigned integer inputs from the user
num1 = int(input("Enter first unsigned integer: "))
num2 = int(input("Enter second unsigned integer: "))
# Ensure inputs are non-negative (Python doesn't have explicit 'unsigned int')
if num1 < 0 or num2 < 0:
   print("Error: Please enter only non-negative integers.")
else:
   # Perform bitwise operations
   and_result = num1 & num2  # Bitwise AND
   or_result = num1 | num2   # Bitwise OR
   xor_result = num1 ^ num2  # Bitwise XOR
   # Print results
   print(f"Bitwise AND: {num1} & {num2} = {and_result}")
   print(f"Bitwise OR: {num1} | {num2} = {or_result}")
   print(f"Bitwise XOR: {num1} ^ {num2} = {xor_result}")
```

5)Write a C program that takes two integers as input and prints their sum, difference, product, and quotient.

Algorithm:

1. Start
2. Declare two integer variables
3. Prompt the user to enter two integers
4. Read the input values
5. Calculate:
    - Sum (+)
    - Difference (-)
    - Product (*)
    - Quotient (/)
6. Print the results
7. End

C Program:

```
#include <stdio.h>

int main() {
   int num1, num2;
   // Taking input from user
   printf("Enter first integer: ");
   scanf("%d", &num1);
   printf("Enter second integer: ");
   scanf("%d", &num2);

   // Performing operations
   int sum = num1 + num2;
   int difference = num1 - num2;
   int product = num1 * num2;
```

```
    // Handling division by zero
    float quotient = (num2 != 0) ? (float)num1 / num2 : 0;

    // Displaying results
    printf("Sum: %d\n", sum);
    printf("Difference: %d\n", difference);
    printf("Product: %d\n", product);

    if (num2 != 0)
        printf("Quotient: %.2f\n", quotient);
    else
        printf("Quotient: Undefined (division by zero not allowed)\n");

    return 0;
}
```

Explanation:

- **Reads two integers** using scanf()
- **Performs arithmetic operations** (addition, subtraction, multiplication)
- **Handles division by zero** by checking if num2 != 0 before division
- **Prints the results** with appropriate formatting

6)Write a C program that takes a character as input and prints its ASCII value.

Algorithm:

1. Start
2. Declare a char variable
3. Prompt the user to enter a character
4. Read the input character
5. Convert the character to its ASCII value using typecasting
6. Print the ASCII value
7. End

C Program:

```
#include <stdio.h>

int main() {
   char ch;
   // Taking input from user
   printf("Enter a character: ");
   scanf("%c", &ch);
   // Printing ASCII value
   printf("ASCII value of '%c' is %d\n", ch, (int)ch);
   return 0;
}
```

Explanation:

- **Reads a character** using scanf("%c", &ch);
- **Converts it to ASCII** using (int)ch
- **Prints the ASCII value**

7)Write a C program that takes a floating-point number as input and prints it with 2 decimal places.

Algorithm:

1. Start
2. Declare a float variable
3. Prompt the user to enter a floating-point number
4. Read the input number
5. Print the number formatted to **2 decimal places** using printf()
6. End

C Program:

```
#include <stdio.h>

int main() {
   float num;
   // Taking input from the user
   printf("Enter a floating-point number: ");
   scanf("%f", &num);
   // Printing the number with 2 decimal places
   printf("Formatted Number: %.2f\n", num);
   return 0;
}
```

Explanation:

- **Reads a floating-point number** using scanf("%f", &num);
- **Formats output with 2 decimal places** using %.2f in printf()

Example Output:

```
Enter a floating-point number: 12.3456
Formatted Number: 12.35
```

8)Write a C program that takes two integers as input and swaps their values without using a third variable.

Algorithm:

1. Start
2. Declare two integer variables
3. Prompt the user to enter two integers
4. Read the input values
5. Swap the values using arithmetic operations:
    - a = a + b
    - b = a - b
    - a = a - b
6. Print the swapped values
7. End

C Program:

```
#include <stdio.h>

int main() {
   int a, b;
   // Taking input from the user
   printf("Enter first integer: ");
   scanf("%d", &a);
   printf("Enter second integer: ");
   scanf("%d", &b);
   // Swapping without using a third variable
   a = a + b;
   b = a - b;
   a = a - b;
   // Printing swapped values
   printf("After swapping:\n");
```

```
    printf("First integer: %d\n", a);
    printf("Second integer: %d\n", b);
    return 0;
}
```

Explanation:

- **Uses arithmetic operations** to swap values without a temporary variable.
- **Works for both positive and negative numbers.**

Example Output:

```
Enter first integer: 5
Enter second integer: 10
After swapping:
First integer: 10
Second integer: 5
```

9)Write a C program that takes an integer as input and prints whether it is even or odd.

Algorithm:

1. Start
2. Declare an integer variable
3. Prompt the user to enter an integer
4. Read the input value
5. Check if the number is divisible by 2:
    - If num % 2 == 0, print "Even"
    - Else, print "Odd"
6. End

C Program:

```
#include <stdio.h>

int main() {
   int num;

   // Taking input from the user

   printf("Enter an integer: ");

   scanf("%d", &num);


   // Checking even or odd

   if (num % 2 == 0)

      printf("%d is Even.\n", num);

   else

      printf("%d is Odd.\n", num);


   return 0;

}
```

Explanation:

- Uses the **modulus operator (%)** to check divisibility by 2.
- If num % 2 == 0, it prints "Even"; otherwise, it prints "Odd".

Example Outputs:

Enter an integer: 8

8 is Even.

Enter an integer: 15

15 is Odd.