CHAPTER-7

DATA HANDLING

TYPE A : VERY SHORT ANSWER QUESTIONS

1.	**What are the main types of C++ data types? Write data types of each type.**
Ans.	C++ offers two types of data types: 1. Fundamental data types: These are five fundamental data types: char, int, float, double and void. 2. Derived data types: These are derived data types: array, function, pointer, reference, constant, class, structure, union and enumeration.
2.	**What do you mean by fundamental data types? How many fundamental data types does C++ provide?**
Ans.	Fundamental data types are the data types that are not composed of any other data type. C++ provides these five fundamental data types: char, int, float, double and void.
3.	**The data type char is used to represent characters. Then why is it often termed as an integer type?**
Ans.	The memory implementation of char data type is in terms of the number code. Therefore, it is said to be another integer data type.
4.	**How are the following statements in the same set different?** **(i) char pcode = 75; (ii) char pcode = 75; (iii) const int a = 5;** **char pcode = 'k' short pcode = 75; const a = 5;**
Ans.	(i) The first statement stores the ASCII value of character 'k', whereas the second statement stores a character 'k'. (ii) The first statement stores the ASCII value of character 'k', whereas the second statement stores numeric value 75. (iii) No difference. In constant declaration when data type is skipped, by default it is *int*.
5.	**If value is an identifier of int type and is holding value 200, is the following statement correct?** **char code = value;**
Ans.	No, the statement is not correct as it stores same garbage value.
6.	**What are the advantages of floating-point numbers over integers?**
Ans.	Following are the advantages of floating-point numbers over integers: 1. The values in between two integers can easily be represented e.g., all possible values between 1 and 2 such as 1.2, 1.313 etc. can be represented. 2. Much larger range of values can be represented using floating-point numbers.
7.	**How many ways can a floating-point be written into?**
Ans.	A floating-point can be written into following two forms: 1. Fractional form 2. Exponent form
8.	**Write 147.9205 into exponent form and 2.4901E04 into fractional form.**
Ans.	(i) 147.9205 = 0.1479205 x 10^3 = 0.1479205E03 (ii) 2.4901E04 = Try at least this...
9.	**Suppose value is an identifier of int type having value 25. After the statement** **float chks = value** **what does the variable chks store?**
Ans.	The variable chks stores 25.
10.	**Why is float type preferred for real quantities? When should the type double be preferred over the shorter type float?**
Ans.	A float type is represent real numbers and have a fractional part, thus it is preferred for real quantities. The double should be preferred over the shorter type float when float is too small or insufficiently precise.
11.	**The data type double is another floating-point type. Then why is it treated as a distinct data type?**
Ans.	The data type double is treated as a distinct data type because, it occupies twice as much memory as type float, and stores floating-point numbers with much larger range and precision.
12.	**"The floating-point numbers should always be stored in the largest type long double because it saves us from the headache of consulting the minimal ranges these data types support". Comment upon this statement.**
Ans.	As we store the floating-point numbers in the largest type long double which has maximum minimal range, we would not have to take tension about available memory space and we can concentrate on our work.

13.	**What is the use of void data type?**
Ans.	The void data type is used as the return type for functions that do not return a value.
14.	**Why does C++ have type modifiers?**
Ans.	C++ has type modifiers to alter the meaning of the base type to fit various situations more precisely.
15.	**What is wrong with the following C++ statement: long float x: ?**
Ans.	In above statement the data type is incorrect. It should be float or long double.
16.	**Using a 7-byte unit, how many different values can be represented on computer?**
Ans.	Using a 7-byte unit, 7.205759403792794e+16 (2^{56}) different values can be represented on computer
17.	**On a machine, short integer is 2 bytes long. What can be the possible size for int integer, short integer and long integer on the same machine, keeping in mind C++ standard?**
Ans.	The possible size for int integer, short integer and long integer can be 2 bytes, 2 bytes and 4 bytes respectively.
18.	**If short integer on a machine is 4 bytes long, what will be the size of unsigned short on the same machine?**
Ans.	If short integer on a machine is 4 bytes long, then the size of unsigned short on the same machine will be 4 bytes.
19.	**What are the variations in char type? How are the variations of char type different from one another?**
Ans.	The variations in char type are: char, unsigned char and signed char. The char and signed char can represent values -128 to 127 whereas, unsigned char can represent values 0 to 255.
20.	**What are the variations in floating-point types? How are the variations of floating-point different from one another?**
Ans.	Following table represent variations in floating-point type and their differences:

Variation Type	Size in bytes	Minimal Ranges
float	4	3.4×10^{-38} to $3.4 \times 10^{38} - 1$
double	8	1.7×10^{-308} to $1.7 \times 10^{308} - 1$
long double	10	3.4×10^{-4932} to $1.1 \times 10^{4932} - 1$

21.	**Arrange the following data types from smallest to largest:** **float, char, double, long double, long, short, int.**
Ans.	char, short, int, long, float, double, long double
22.	**Explain the difference among 0, '0', '\0', and "0".**
Ans.	0: is a integer constant. '0': is a character constant. '\0': is a escape sequence for Null. "0": is a string constant.
23.	**What is the distinction between 33L and 33?**
Ans.	An l or L suffix indicates it is a long integer constant. Thus 33L is a long integer constant and 33 is an integer constant.
24.	**Write the statements you would use in C++ to find out which character the code 86 represents.**

Ans.	`void main()`
	`{`
	`  char value=86;`
	`  cout<<"Character="<<value;`
	`}`

25.	**What do you understand by the following:** **(i) array (ii) structure (iii) function** **(iv) pointer (v) union (vi) enumeration**
Ans.	Same as answer of Q No. 1 of Long Answer Questions.
26.	**What is a reference variable? How is it defined in C++?**
Ans.	A reference variable provides an alias for a previously defined variable. It is defined in C++ with following syntax: type &ref-var = var-name; where type is any valid C++ data type, ref-var is the name of reference variable that will point to variable denoted by var-name.
27.	**What is wrong with the following statement?** **const int y;**

Ans.	A constant must be initialized at the time of declaration. Hear, the constant variable y is not initialized which is invalid.
28.	**What do you understand by a class in C++? What does a structure represent in C++?**
Ans.	A class is collection of similar objects. A structure represents the data elements in C++.
29.	**What are the similarities and difference between an array and a structure?**
Ans.	Similarities: ✓ Both are collection of elements. Differences: ✓ An array can hold multiple elements of same data type whereas a structure can hold multiple elements of different data types. ✓ The component element of an array is referenced by the array name and the index value e.g., NAME[3], FILE[7] etc. on the other hand, the component element of a structure is referenced by the structure name and the elemet name e.g., style.main, check.value etc.
30.	**What are the similarities and difference between a class and a structure?**
Ans.	Similarities: ✓ Both are user defined data types and store elements of different data types. Differences: ✓ A class can represent data elements as well as their associated functions that manipulate them whereas a structure can only represent the data elements. It has no control over the associated functions of data.
31.	**Given the following definitions:** **int inal = 1024;** **char ch = 70;** **int i = 'a';** **const int ic = i;** **Which of the following statements are illegal? Why?** **(i) i = ic;, (ii) ic = i; (iii) ch = i;** **(iv) i = ch; (v) ch = ival; (vi) ch = '10';**
Ans.	(ii) ic = i is illegal, as constant variable cannot be modify. (iii) ch = i is illegal, as it initialize a char variable with int variable. (iv) i = ch is illegal, as it initialize a int variable with char variable. (v) ch = ival is illegal, as variable ival is undefined.

32.	**Find errors, if any in the following C++ statements:**
	`(i) int code = three; // three is an enumerator`
	`(ii) enum (gree, yellow, red);`
	`(iii) const int size;`
	`(iv) size = 10; //size is already defined constant.`

Ans.	(i) It is valid only when enumerations are treated as integers. (ii) Wrong brackets. It should be: `enum {gree, yellow, red};` (iii) A constant variable must be initialized at the time of declaration. (iii) No, we can't assign a value to a constant variable.
33.	**What are derived data types? Name the user defined data types in C++.**
Ans.	The data types that are composed of fundamental data type are called **derived data types**. Following are the user defined data types: 1. Class 2. Structure 3. Union 4. Enumeration
34.	**What is a variable? In C++, two values are associated with a symbolic variable. What are these?**
Ans.	A variable is a named storage location, whose values can be manipulated during program run. Following are the two values which are associated with a symbolic variable: (i) rvalue that gives its contents. (ii) lvalue that gives its address.
35.	**Why is a variable called symbolic variable?**
Ans.	The variables are called symbolic variables because these are named locations. For example, the following

	statement declared a variable *i* of the data type int: `int i;`
36.	**Write declaration for** **(i) a pointer to a character and an array of 10 integers.** **(ii) a reference to a floating-point variable.** **(iii) a constant character and a constant integer.**

Ans.	`(i) char *ptr;`
	`int a[10];`
	`(ii) float total;`
	`float &sum = total;`
	`(iii) const char name='A';`
	`const int MAX=50;`

37.	**Define the following: (i) A short integer with the value 90. (ii) An unsigned int integer with the value 31240. (iii) an integer with the values 3000000000.**

Ans.	`(i) short int a=90;`
	`(ii) unsigned int var=31240;`
	`(iii) int b=3000000000;`

38.	**What do you mean by dynamic initialization of a variable? Give an example.**
Ans.	In dynamic initialization a variable can be initialized at run time using expressions at the place of declaration. For example, `float avg = sum/count;`
39.	**What is the impact of access modifier const over a variable?**
Ans.	The access modifier const modifies the access type of variable, the variable becomes constant and its value remains unchanged throughout the program, it can never be changed during program run.

TYPE B : SHORT ANSWER QUESTIONS

1. What is the difference between fundamental data types and derived data types? Explain with examples.

Ans.

Fundamental data types	Derived data types
Fundamental data types are not composed of any other data type.	**Derived data types** are composed of fundamental data type.
These are five fundamental data types: char, int, float, double and void.	These are derived data types: array, function, pointer, reference, constant, class, structure, union and enumeration.
Example: `int a,b;`	Example: `int arr[10];`

2. Explain fundamental data types and their purpose.

Ans. Same as answer of Q No. 1 of Long Answer Questions.

3. What main integer types are offered by C++?

Ans. Following are the main integer types offered by C++:
short, unsigned short, signed short, int, unsigned int, signed int, long, unsigned long, signed long

4. When are unsigned integers preferred over signed integers? What is the advantage of unsigned integers over signed integers?

Ans. The unsigned integers are preferred over signed integers for quantities that are never negative such as populations, sports scores, inventory counts etc. The advantage of unsigned integers over signed integers is it increases the largest value the variable can hold.

5. Explain floating point types offered by C++ along with the minimal ranges and digits of precision they support.

Ans.

Variation Type	Size in bytes	Minimal Ranges	Digits of precision
float	4	3.4×10^{-38} to $3.4 \times 10^{38} - 1$	7
double	8	1.7×10^{-308} to $1.7 \times 10^{308} - 1$	15
long double	10	3.4×10^{-4932} to $1.1 \times 10^{4932} - 1$	19

6. Explain the concept of a pointer and its purpose.

Ans. A pointer is a variable that holds a memory address of another variable. If one variable contains the address of another variable, the first variable is said to point to the second. The general form of declaring a pointer variable is
type *ptr;

	where *type* is any valid C++ type and *ptr* is the name of the pointer variable e.g. char *a; declares a pointer to a character.
7.	**What is the significance of a reference variable? Explain with example.**

Ans.	A reference variable provides an alias for a previously defined variable. For example, if avg is declared as a reference variable for a variable percentage, then both avg and percentage represent the same variable and can be used interchangeably. Following code illustrate it:
	`int percentage;`
	`int &avg = percentage;`

8.	**Explain with example the concept of constant variable.**

Ans.	A variable whose value can never change during the program run is called constant variable. The general form of constant declaration is as follows:
	`const type name=value;`
	where *const* is a keyword, *type* is any valid C++ data type, *name* is the name of the constant and *value* is the constant value of the data type *type*. For example,
	`const int MIN=10;`
	A constant must be initialized at the time of declaration.

9.	**Write declaration for a class that holds properties of a branch. The class holds information like branch-number, area, number of employees, head and the operations associated like branch report printing, branch data modification, pending works reporting, forecast reporting.**

Ans.	`class Branch`
	`{ int branch_number;`
	`char area[30];`
	`int Num_of_emp;`
	`char Head[20];`
	`public:`
	`void Print_Report();`
	`void Modify_Data();`
	`void Pending_Report();`
	`void Forecast_Report();`
	`};`

10.	**Write a declaration for a structure that holds same information as that of the class mentioned in question 9.**

Ans.	`struct Branch`
	`{ int branch_number;`
	`char area[30];`
	`int Num_of_emp;`
	`char Head[20];`
	`};`

11.	**Write a declaration for a structure that holds information of a student like roll no, name, class, marks and grade.**

Ans.	`struct Student`
	`{ int Rollno;`
	`char ame[20];`
	`int class;`
	`float marks;`
	`char grade;`
	`};`

12.	**Write declaration for a class that holds the information of Q. 11. For associated operations, make suitable assumptions.**

Ans.	`class Student`
	`{ int Rollno;`
	`char ame[20];`
	`int class;`
	`float marks;`
	`char grade;`
	`public:`

```
        void getdata();
        void Display();
        void cal_grade();
};
```

13. **Explain the function and usage of a union giving an example.**

Ans. A union represents a memory location shared by two or more different variables. The keyword union is used for declaring and creating a union. For example,

```
union share {
                int i;
                char ch;
              };
union share cnvt;
```

Above example declares a union *share* having two variables and creates a union object *cnvt* of union type *share*. In union *cnvt*, both integer *i* and character *ch* share the same memory location.

14. **Why are so many data types provided in C++?**

Ans. The reason for providing so many data types is to allow programmer to take advantage of hardware characteristics. Machines are significantly different in their memory requirements, memory access times, and computations speeds. Suppose, a program that earlier was working with 2 byte *int* efficiently, on shifting to a machine that provides 4-byte *int* and 2-byte *short* can easily work with short of the new machine thereby not increasing the memory requirements of the program.

15. **Explain the function and usage of variables with examples.**

Ans. Variables represent named storage locations, whose values can be manipulated during program run. For example, to store name of a student and marks of a student during a program run, we require storage locations that too named so that these can be distinguish easily. The variables are called symbolic variables because these are named locations. For example, the following statement declared a variable *var* of the data type *int*:

```
int var;
```

It has two associated values: rvalue that gives its contents and lvalue that gives its address.

16. **How many ways can a variable be initialized into? Give examples for each type of initialization.**

Ans. A variable can be initialized into following two ways:

1. Syntax: `type var-name = value;`
   Example: `int val = 1001;`
2. Syntax: `type var-name (value);`
   Example: `int val (1001);`

In both the cases, *val* is initialized with a first value of 1001.

17. **Explain the impact of access modifier const variables. Support your answer with examples.**

Ans. Same as answer of Q No. 39 of Very Short Answer Questions. Example is as following:

```
void main()
{
      const int MAX=50;     //constant variable
      int a=MAX;
      cout<<a;
      a=10;
      cout<<a;
}
```

In above code value of the constant variable *MAX* cannot be change, if we try to change it, an error will be occurred, whereas value of variable *a* can be change during program execution.

18. **"Comments are useful and easy way to enhance readability and understandability of a program" Elaborate with examples.**

Ans. Comments are pieces of code that the compiler discards or ignores or simply does not execute. The purpose of comment is only to allow the programmer to insert some notes or descriptions to enhance readability or understandability of the program. For example,

```
//A sample program          <-->  Single line comments
#include<iostream.h>              // includes header files
```

```
int main()
{     cout<<"Hello There!!";
            /* This is a simple program with just one output statement
               written for the purpose of explaining comments */
}
```

Multiline comment

19. **Write a short program that asks for your height in centimeters and then converts your height to feet and inches. (1 foot = 12 inches, 1 inch = 2.54 cm.).**

Ans.

```
#include<iostream.h>
#include<conio.h>
void main()
{
      const float CM_TO_INCH = 2.54;
      const float INCHE_TO_FEET = 12;
      const float CMS_TO_FEET = CMS_PER_INCH * INCHES_PER_FEET;
      cout << "Input a value in centimeters you wish to convert to feet and
inches: ";
      float cm = 0;
      cin >> cm;
      int feet = int (cm / CMS_TO_FEET);
      float inch = (cm / CMS_PER_INCH) - (feet * INCHES_PER_FEET);
      cout << cm << " centimeters is " << feet << " feet " << inch << "
inches." << endl;
      getch();
}
```

20. **An electricity board charges according to following rates:**
For the first 100 units – 40 P per unit (P – Paise)
For the next 200 units – 50 P per unit
Beyond 300 unit – 60 P per unit.
All users are charged meter charge also which is Rs. 50/-.
Write a program to read the names of users and numbers of units consumed, and print out the charges with names.

Ans.

```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
void main()
{     char name[20];
      int unit;
      float charge;
      clrscr();
      cout<<"Enter name: ";
      gets(name);
      cout<<"Enter unit: ";
      cin>>unit;
      if(unit<=100)
            charge=((unit*40)/100)+50;
      else if(unit>100 && unit<=300)
            charge=((unit*50)/100)+50;
      else(unit>300)
            charge=((unit*60)/100)+50;
      cout<<"Name: "<<name<<endl;
      cout<<"Total Charge: "<<"Rs. "<<charge<<" including meter charge of
Rs.50";
      getch();
}
```

21. **Write a program to read a number n and print n^2, n^3, n^4, and n^5.**

Ans.

```
#include<iostream.h>
#include<conio.h>
void main()
{     int n;
      clrscr();
      cout<<"Enter n: ";
      cin>>n;
      cout<<n*n<<"\t"<<n*n*n<<"\t"<<n*n*n*n<<"\t"<<n*n*n*n*n;
      getch();
}
```

22. **Write a program to find area of a triangle.**

Ans.

Formula

Area = ½ × b × h

b = base

h = vertical height

```
#include<iostream.h>
#include<conio.h>
float area(float n,float b,float h)
{
    float ar;
    ar=n*b*h;
    return ar;
}
void main()
{
    float b,h,r,l;
    float result;
    clrscr();
    cout<<"\nEnter the Base & Hieght of Triangle: \n";
    cin>>b>>h;
    result=area(0.5,b,h);
    cout<<"\nArea of Triangle: "<<result<<endl;
    getch();
}
```

23. **Write a program to find whether a given number is even or odd.**

Ans.

```
#include<iostream.h>
#include<conio.h>
void main()
{     int n;
      clrscr();
      cout<<"Enter n: ";
      cin>>n;
      if(n%2==0)
            cout<<"The number is even";
      else
            cout<<"The number is odd";
      getch();
}
```

24. **Write a program to accept three digits (i.e., 0 – 9) and print all possible combinations from these digits. (For example, if the three digits are 1, 2, and 3 then all possible combinations are 123, 132, 231, 213, 312 and 321).**

Ans.

```
#include< iostream.h >
#include< conio.h >
void main( )
{
```

```
   int a[3];
   clrscr( );
   cout<<"Enter three digits :"<
   cin>>a[0]>>a[1]>>a[2];
   for ( int i = 0; i< 3; i++ )
   {
     for( int j = 0; j < 3;  j++ )
     {
        for ( int k = 0; k < 3;  k++ )
        {
           if ( i! = j && i! = k && j! = k )
             cout << endl << endl << a[ i ] << a[ j ] << a[ k ];
        }
     }
   }
   getch( );
}
```

25. **Write a program to convert given inches into its equivalent yards, feet and inches. (1 yard = 36 inches, 1 foot = 12 inches)**

Ans.

```
#include <iostream.h>
#include <conio.h>
void main( )
{
  int in, yrd, ft, r;
  cout << " Enter distance/length in inches ";
  cin >> in;

  yrd = in/ 36;
  r = in % 36;
  ft = r / 12;
  r = r % 12;

cout << " \n The distance in inches is " << in ;
cout << " \n The yard-feet-inches = " << yrd << " yard" << ft << "feet" << r
<< "inches" ;
getch( );
}
```

26. **Write a program to read two numbers and print their quotient and remainder.**

Ans.

```
#include<iostream.h>
#include<conio.h>
void main()
{     int n1,n2,qut,rem;
      clrscr();
      cout<<"Enter n1: ";
      cin>>n1;
      cout<<"Enter n2: ";
      cin>>n2;
      qut=n1/n2;
      rem=n1%n2;
      cout<<"Quotient: "<<qut<<endl<<"Reminder: "<<rem;
      getch();
}
```

27. **Write a program to compute simple interest and compound interest.**

Ans.

```
#include<iostream.h>
#include<conio.h>
#include<math.h>
```

```
void main()
{     float p,r,n,i,a;
      clrscr();
      cout<<"Enter the principle amount: ";
      cin>>p;
      cout<<"Enter the rate of intrest: ";
      cin>>r;
      cout<<"Enter the duration: ";
      cin>>n;
      i=p*r*n/100;
      a=p*pow((1+r/100),n);
      cout<<"Simple Intrest: "<<i<<endl<<"Compound Intrest: "<<a;
      getch();
}
```

28. **Write a program to: (i) print ASCII code for a given digit. (ii) print ASCII code for backspace.**
(Hint: Store escape sequence for backspace in an integer variable).

Ans.

```
(i) #include<iostream.h>
    #include<conio.h>
    void main()
    {
      clrscr();
      char ch;
      int num;
      cout<<"Enter digit: ";
      cin>>ch;
      num=ch;
      cout<<"The ASCII code for "<<ch<<"is "<<num<<endl;
      getch();
    }

(ii)#include<iostream.h>
    #include<conio.h>
    void main()
    {clrscr();
      char ch='\b';
      int num=ch;
      cout<<"The ASCII code for backspace is "<<num<<endl;
      getch();
    }
```

TYPE C : LONG ANSWER QUESTIONS

1. **Explain C++ data types with appropriate examples.**

Ans. Data types are means to identify the type of data and associated operations of handling it. C++ offers two types of data types:

1. Fundamental data types: These are the data types that are not composed of any other data type. Following are the five fundamental data types:

(a) char: Stores character. For example, ‘a’, 65

(b) int: Integers are whole numbers. For example, 5, 39, -1917, 0

(c) float: A number having fractional part is a floating-point number. For example, 3.14159

(d) double: Same as float data type but occupies twice as much memory as type float. For example, 5.631

(e) void: Specifies an empty set of values.

2. Derived data types: These are the data types that are composed of fundamental data type. Following are the derived data types:

(a) Array: Array is a collection of element that has same name and same data type. For example, int a[10].

(b) Function: A function is a named part of a program that can be invoked from other parts of the program as often

needed. For example, float cube (float);.
(c) Pointer: A pointer is a variable that holds a memory address. For example, char *a;.
(d) Reference: A reference is an alternative name for an object. For example, int &sum = total;.
(e) Constant: A constant is a data item whose data value can never change during the program run. For example, const int SIZE = 50;.
(f) Class: A class represents a group of similar objects.
(g) Structure: A structure is a collection of variables of different data types referenced under one name.
(h) Union: A union represents a memory location shared by two or more different variables.
(i) Enumeration: Enumeration is an alternative method for naming integer constants.

2. **Explain user-defined data types in C++ with appropriate examples.**

Ans. The data types which are defined by users are called user-defined data types. Following are user-defined data types:

1. Class: A class represents a group of similar objects which is achieved by keyword class. For example,

```
class Student
{
      char name[20];
      int rollno;
      float mark;
   public:
      void getdata();
      void putdata();
};
```

2. Structure: A structure is a collection of variables of different data types referenced under one name, providing a convenient means of keeping related information together which is achieved by keyword struct. For example,

```
struct department
{
      char name[20];
      int num_emp;
      char h_o_d[20];
      float salary;
};
department newemp;
```

3. Union: A union represents a memory location shared by two or more different variables, generally of different types at different times which are achieved by keyword union. For example,

```
union first {
                int a;
                char c;
              };
union first f1;
```

4. Enumeration: Enumeration is an alternative method for naming integer constants which is achieved by keyword enum. For example,

```
enum status { START = 0, PAUSE, GOO };
```

3. **Explain the difference between a variable, reference variable, a constant variable and a pointer variable. Support your answer with examples.**

Ans.

Variable	Reference variable	Constant variable	Pointer variable
A variable is a named storage location. It has two associated values: rvalue that gives its contents and lvalue that gives its address.	A reference variable provides an alias for a previously defined variable.	A constant variable is a variable whose data value can never change during the program run. For example,	A pointer is a variable that holds a memory address of another variable.
Example: Int a = 10;	Example: Int &per = avg;	Example: const int MAX = 5;	Example: int *ptr;

4. **What is the impact of type modifiers on base data types? Support your answer with examples for each data type.**

Ans. When the data type modifiers appear before a data type, the meaning of the data type is changed in the sense that

	its size is affected thereby affecting minimal range of values the data type can represent. Following are the examples for each data type: 1. Integer Type Modifiers: For instance, an int is by default 2 bytes long and hence can represent values -32768 to 32767 but when modifier unsigned appears before it, its range becomes 0 to 65,535. 2. Character Type Modifiers: For instance, a char is by default 1 bytes long and hence can represent values -128 to 127 but when modifier unsigned appears before it, its range becomes 0 to 255. 3. Floating Type Modifiers: For instance, a float is by default 4 bytes long and hence can represent values 3.4×10^{-38} to $3.4 \times 10^{38} - 1$ but when modifier long appears before it, its range becomes 3.4×10^{-4932} to $1.1 \times 10^{4932} - 1$.