1. With suitable examples differentiate between run-time error and logical error.

**Ans:**

|  |  |
| --- | --- |
| **Run time error** | **Logical error** |
| Syntactically correct statement performs illegal operation because program encounters unexpected data during execution of a program is called Run-Time errors. | An error in program design or program implementation that does not prevent your program from compiling, but causes it to do something unexpected. |
| Example:  Division by zero (0),  Square root of a negative number  Logarithm of 0 or negative number | Example:  Accumulator and counter not initialized  Incorrect formulae  Incorrect use of operators |

1. With suitable examples differentiate between run-time error and syntax error.

**Ans:**

|  |  |
| --- | --- |
| **Syntax error** | **Run time error** |
| Error committed when the grammar of the language is violated. | Syntactically correct statement performs illegal operation because program encounters unexpected data during execution of a program is called Run-Time errors. |
| Syntax errors are detected at compile time. | Run time errors are detected at run time. |
| Example:  Typographical mistakes  Using a variable without creating  Call to undefined function | Example:  Division by zero (0),  Square root of a negative number  Logarithm of 0 or negative number |

1. With examples differentiate between entry controlled loop and exit controlled loop.

**Ans:**

|  |  |
| --- | --- |
| **Entry controlled Loop** | **Exit Controlled Loop** |
| Looping condition is checked at the beginning of the loop. | Looping condition is checked at the end of the loop. |
| Loop does not execute even once if the looping condition is false in the beginning itself. | Loop executes at least once irrespective of the looping condition. |
| Example:  for (int x=2; x>5; x++)  {  cout<<x<<", ";  } | Example:  int x=2;  do  {  cout<<x<<", ";  }while (x>5); |

1. With examples mention any two differentiate between local variable and global variable.

**Ans:**

|  |  |
| --- | --- |
| **Local Variable** | **Global variable** |
| Default value of a Local Variable is **garbage** | Default value of a Global Variable is **0** |
| A local variable is **visible inside** the block and blocks nested blow | A global variable is **visible throughout** the program – main() function and all other user defined functions |
| **Longevity** of a local variable is as long as the **block is active** | **Longevity** of a global variableis as long as the **program is active** |
| Example:  #include <iostream.h>  int a=4; // global variable  void first()  {  int b = a; // b is local variable  cout<<(a+b)<<endl;  a++;  b--;  }  void main()  {  cout<<a; //global a  first(a);  cout<<b; //Syntax error, b created in first() cannot be used in main()  } | |

1. With examples write any two differences between actual parameter and formal parameter.

**Ans:**

|  |  |
| --- | --- |
| **Actual Parameter** | **Formal parameter** |
| It is the parameter used in function invocation | It is the parameter used in the header of function definition |
| Actual parameter may be a variable or an expression or a constant | Formal parameter is always variable (or an alias) |
| Example:  #include <iostream.h>  void add(int a, int b) //int a, int b are formal parameters  {  int s=a+b;  cout<<s<<endl;  }  void main()  {  int x=10, y=20;  add(x, y); //x, y are actual parameters  add(x+5, y+15); //x+5, y+15 are actual parameters  } | |

1. Write any three differences between value parameter and reference parameter.

**Ans:**

|  |  |
| --- | --- |
| **Value Parameter** | **Reference Parameter** |
| Copy of actual parameter | Alias of actual parameter |
| Change in value parameter does not change actual parameter | Change in reference parameter, updates actual parameter |
| Transfer of data is one way, from calling function to called function | Transfer of data is two ways, from calling function to called function and vice-versa |
| Example:  #include <iostream.h>  void paratry(int a, int &b) //int a value parameter  { //int &b reference parameter  a+=b;  b+=a;  cout<<a<<","<<b<<endl;  }  void main()  {  int x=10, y=20;  paratry(x, y);  cout<<x<<","<<y<<endl;  }  Running of the program produces following output:  30,50  10,50  Value parameter int a does not update actual parameter x but reference parameter int &b updates actual parameter y | |

1. With suitable example differentiate between function declaration/prototype and function definition.

**Ans:**

|  |  |
| --- | --- |
| **Function Declaration/Prototype** | **Function Definition** |
| A function declaration contains Function name, Return type of the function, optional list of formal parameters, and a Semi-colon at the end. | A function definition is the complete function, that is, header and the body. |
| Name of the formal parameters are not compulsory in a function declaration. | Name of the formal parameters are compulsory in a function definition. |
| Example:  #include<iostream.h>  double factorial(int); //function declaration  void main()  {  int m;  cout<<"Input an integer? ";  cin>>m;  double f=factorial(m);  cout<<m<<"!="<<f<<endl;  }  double factorial(int n)  {  Function definition  double fact=1;  for (int k=1; k<=n; k++)  fact\*=k;  return fact;  } | |

1. Name the fundamental data types of C++ used for creating a variable.

**Ans:** char, int, double, float.

1. Name all the fundamental data types of C++ for which memory is allocated.

**Ans:** char, int, double, float.

1. Name the type modifiers of C++. Name the data types that do not support any type modifiers.

**Ans:**

Type modifiers of C++ are: short, long, signed, unsigned

void and float don’t support any type modifers.

1. Name the data types that support two or more type modifiers.

**Ans:**

int and char. int supports all the type modifiers and char supports signed and unsigned.

1. Identify the data types of the expressions/constants given below:

i) 9 + '9' ii) -12.5+56 iii) "9" iv) '0'

**Ans:** i) int ii) float/double iii) string iv) char

1. What is a C++ comment? What are two ways of writing C++ comment?

**Ans:**

A C++ comment is a non-executable statement in the program. It is ignored by the compiler and no code is generated for it.

Two ways of writing C++ comments:

(i) Single line Comment: Single Line Comment starts with pair of forward slash (//) and till the end of line

//comment in C++ style

(ii) Multi-line comment: Multi-line comment start with forward slash and star (/\*) and ends with star and forward slash (\*/). Example:

/\*

multi-line comments

comment in C style

\*/

1. What data type will you use to declare the following variables:
2. To store the Section of a student.
3. To store the population of a country which may run into billions.
4. To store the distance travelled by light in *n* seconds.

**Ans:** (i) char (ii) unsigned long int (iii) double/float

1. In context of C++ differentiate between *a* and ‘*a*’.

**Ans:**

|  |  |
| --- | --- |
| **a** | **‘a’** |
| Identifier | Character constant |
| Memory occupied depends on the data type | Occupies 1 byte |

1. What relational operators are available in C++.

**Ans:** >, >=, <, <=, !=, ==

1. Write C++ statements for the following:

i) To divide an integer variable *a* by another integer variable *b* and store the result in *a*.

ii) Decrease the value of a float variable *amount* by 1 without using assignment operator.

iii) All possible ways of calculating the square root of a floating point variable x and storing the result in another floating point variable b.

**Ans:**

i) a/=b; (OR a=a/b;)

ii) amount--; (OR --amount)

iii) b=sqrt(x); b=pow(x, 0.5);

1. What is typecasting? What are two ways of typecasting in C++?

**Ans:**

Converting data from one type to another type temporarily, inside the processor (CPU) is called typecasting. Two ways of typecasting in C++ are:

1. Keeping the operand in parentheses. Example:

int n = int('A');

1. Keeping the data type in parentheses. Example:

int n = (int) 'A';

1. Name two operators that work as unary and as well as binary operator.

**Ans:** +, -

1. Define token. Name different categories of tokens in C++?

**Ans:**

Building block of a program is called a token. It is also called program element.

Categories of tokens in C++: Keyword, Identifier, Constant, Operator, String and Comment.

1. Give two examples each of the following:
2. char constant, (ii) int constant, (iii) float constant (iv) String constant

**Ans:**

(i) ‘a’, ‘9’ (ii) 234, -56 (iii) 2.45, 5.6 (iv) “FAIPS”, “C++”

1. How many bytes of memory will be allocated to variable of the type:

i) **char** ii) **int** iii) **float** iv) **double**

**Ans:** i) 1, ii) 4, iii) 4, iv) 8

1. Name the type modifiers of C++. Name the type modifiers that can be used with data type float and double.

**Ans:** Type modifiers in C++ are: signed, unsigned, short, long.

float does not support any type modifier. double supports long.

1. Identify the data type of the expressions given below:

i)15/4.0 ii)'2'-48 iii)2+3.5\*7 iv)(**int**)12.5/2.5

**Ans:** i) float (or double) ii) int iii) float (or double) iv) float (or double)

1. Name the rules for naming a C++ identifier.

**Ans:** Rules for naming a C++ identifier:

* 1. Variable name should start with an alphabet (letter) or an underscore.
  2. Variable name may contain more than one character. Second characters onwards we may use only alphabets or digit or underscore.
  3. No special characters are allowed in a variable name except underscore.
  4. A variable name in C++ is case sensitive. Uppercase and lowercase letters are distinct.
  5. A variable name cannot be a keyword.

1. Identify three incorrect identifier names and explain why, from the list given below:

long, AD\_No, INT, comp-sc, CAL29, 2ndfloor, price, cell#

**Ans:**

1. long – It is a keyword
2. comp-sc – It uses a special character other than underscore ( - )
3. 2ndfloor – It starts with digit (2)
4. cell# - It uses a special character other than underscore ( # )
5. c) Identify the data type of constants / expressions given below:

i) 30.0/4 ii) "30.4/4" iii) 30/4 iv) '4'

**Ans:**

i) float/double ii) string iii) int iv) char

1. Mention two differences between data type float and data type double.

**Ans:**

i) float uses 4 bytes whereas double uses 8 bytes.

ii) float does not support any type modifier whereas long supports ‘long’ type modifier.

1. Write two differences between keyword and built-in identifier.

**Ans:**

Differences between keyword and built-in identifier

* + 1. A keyword cannot be redefined in a program whereas a built-in identifier can be redefined.
    2. To use a keyword we don’t need to include any header file because keywords are stored in the compiler whereas to use a built-in identifier we need to include correct header file in the program.

1. i) Name any two unary operators other + and –.

ii) Name any two operators that works from right to left other than =.

iii) Name the two operators that are used to combine two or more logical expression.

**Ans:**

1. **++, --**
2. **+=, -=, \*=, /=, %=**
3. **&&, ||**

1. Write four possible C++ statements to decrement the value of a variable x by 1.

**Ans:**

i) x--; ii) –-x; iii) x-=1; iv) x = x-1;

1. Give an example of cascading of output operator.

**Ans:**

int x=10; double y=3.4; char z='D';

cout<<"x="<<x<<", y="<<y<<", z="<<z<<endl;

1. What is conditional operator? Explain with example.

**Ans:**

The operator “?:” is the conditional operator in C++. It is also called Ternary operator and is used in place of if-else statement.

Example:

If-else statement

if (a>b)

max = a;

else max = b;

Ternary operator

**int** max = a>b ? a : b;

**Or,**

int max;

a>b : max=a : max=b;

1. Write C++ logical expression for the following (do not use any C++ built-in functions):

i) To check that a character variable mychar contains only alpabets

ii) To check that an integer variable number is even no not divisible by 4

iii) To check that an integer variable marks contains a value between 300 and 500

iv) To check that a character variable alpha contains lowercase vowel

**Ans:**

**i) mychar >= ‘A’ && mychar <= ‘Z’**

**ii) number%2 == 0 && number%4 != 0**

**iii) marks >= 300 && marks <= 500**

**iv) alpha == ‘a’ || alpha == ‘e’ || alpha == ‘i’**

**|| alpha == ‘o’ || alpha == ‘u’**

1. Write C++ logical expression for the following (do not use any C++ built-in functions):
2. To check that a character variable **mychar** contains only digit.
3. To check that an integer variable **number** is odd but not divisible by 5.
4. To check that an integer variable **marks** contains a value between 0 and 100.
5. To check that a character variable **alpha** contains uppercase vowel.
6. To check that a character variable **ch** contains an **uppercase**.
7. To check that an integer variable **num** is an odd number other than **5**.
8. To check that an integer variable **yr** is divisible by 4 but not by 100.

**Ans:**

i) mychar >= ‘0’ && mychar <= ‘9’

ii) number%2 == 1 && number%5 != 0

iii) marks >= 0 && marks <= 100

iv) alpha == ‘A’ || alpha == ‘E’ || alpha == ‘I’

|| alpha == ‘O’ || alpha == ‘U’

v) ch >= ‘A’ && ch <= ‘Z’

vi) num%2 == 1 && num != 5

1. yr%4 == 0 && yr%100 != 0
2. Name the keywords which are optional in switch-case.

**Ans:** break and default are optional keywords in switch-case.

1. What is an alias? How is an alias created? Give a suitable example to create an alias.

**Ans:**

An alias is another name given to an already existing variable. An alias is created by the following rule:

DataType& NewVariableName=OldVariableName;

Example:

**int** x=35;

**int**& y=x;

1. What are the two parts of a function in C++? Define the role of each part.

**Ans:**

Two parts of a function are Function Header and Function Body.

1. **Function Header**: it contains the **name** of the function, **return type** of the function and **optional** list of formal parameters. Function header is also called **Function Declarator**.
2. **Function Body**: it is the block after the function header. Function body contains statements that carry out action inside the function including the optional **return** statement. If the return value of a user defined function is **void**, then return statement is not required. Function Body is also called **Function Block**.
3. Define global variable. When scope resolution operator is necessary with a global variable.

**Ans:**

A variable created immediately after the header files and before any block is called Global Variable. A global variable may be used in anywhere in the program and it is visible throughout the program.

Inside a block, if a local variable and a global variable have the same name and we want to use global variable then it is necessary to use scope resolution operator with it.

1. Define global variable. Write any two characteristics of global variable.

**Ans:**

A variable created immediately after the header files and before any block is called Global Variable.

Characteristics of Global variable:

a) Default value of a Global Variable is 0

ib) A global variable is visible throughout the program – main() function and all other user defined functions

1. Define local variable. Write any two characteristics of local variable.

**Ans:**

A variable created inside a block is called a Local Variable.

Characteristics of Local variable:

a) Default value of a Local Variable is garbage

b) Visible inside the block and blocks nested blow

1. What happens in the following situations:

i) Return type of a function is double but it does not return any value.

ii) Return type of a function is void but it returns a value.

**Ans:**

i) Compiler flashes a warning and substitutes a random value at function call.

ii) Compiler flashes an error.

1. When is return statement necessary in a C++ function? What is the role of return statement?

**Ans:**

Return statement is necessary in a C++ function when the return type of the function is not void but some other data type.

Role of return statement:

a) It terminates the function

b) Returns a value from called function to the calling function

1. What are the optional parts in a C++ function?

**Ans:**

List of formal Parameters and return statement.

1. What are the two categories of formal parameters?

**Ans:**

Value parameters and Reference parameters.