1. What is a backslash character constant (Escape sequence)? Write any two characteristics of backslash character constants.

**Ans:**

A backslash constant is a special type character constant which start with back slash (\) and is followed by a character. Two characteristics of backslash constants are:

1. Backslash constants are whitespace characters.
2. Backslash constants can be embedded in strings.
3. Give examples of two Escape sequences along with their purpose.

**Ans:**

|  |  |
| --- | --- |
| Escape Sequence | Purpose |
| '\a' | Alert (Bell) character - Makes a beep sound |
| '\b' | Back space - Moves the cursor one column to the left |
| '\n' | New line - Shifts the cursor to the beginning of next line |
| '\r' | Carriage return - Takes the cursor to the beginning of the current line |
| '\t' | Tab character - Cursor jump 8 columns to the right |

*(Write any two of these)*

1. Write output generated by the following C++ program segment:

cout<<"This first li\ne "; //space after e

cout<<"of code \before "; //space after e

cout<<"seco\nd line of st\ring";

**Ans:**

**This first li**

**e of codeefore seco**

**ingine of st**

1. Write output generated by the following C++ program segment:

**void** main()

{

cout<<"MANGAF"<<'\b'<<"KUWAIT\nAHAMDI";

cout<<"\tFAIPS\t"<<"DPS\n";

cout<<"FAIPS\rDPS";

}

**Ans:**

**MANGAKUWAIT**

**AHAMDI FAIPS DPS**

**DPSPS**

1. Write output generated by the following C++ program segment:

void main()

{

cout<<"India\ns in ku\wait";

cout<<"\rIndo-"<<'\n';

cout<<"Educa\tion Boards";

getch();

}

**Ans:**

**India**

**Indo-kuwait**

**Educa ion Boards**

1. Write output generated by the following C++ program segment:

void main()

{ cout<<"\\\\displays\\\n";

cout<<"\\g displays \g";

}

**Ans:**

**\\displays\**

**\g displays g**

1. What is a user defined constant (Read-only identifier)? Also give an example of a user defined constant.

**Ans:**

**It is an identifier whose value cannot be altered in a program.**

**Example: The statement**

**const int MAX = 5;**

**defines MAX to be an integer constant (due to the keyword const) with value 5. The value of MAX remains 5 and cannot be changed throughout the program.**

1. What is a macro? Also give an example of a macro.

**Ans:**

**Macro is an identifier created by using Compiler Directive #define. Macro identifier represents replacement text.**

**Example: The statement**

**#define MAX 5**

**defines MAX to be a macro and 5 is its replacement text.**

1. Write any two differences between Macro Identifier and Constant identifier.

**Ans:**

|  |  |
| --- | --- |
| **Macro Identifier** | **Const identifier** |
| Macro Identifier has **no data type** | Constant Identifier has a **data type** |
| **No memory is allocated** to a Macro Identifier | **Memory is allocated** to a Constant Identifier |
| Compiled code **does not contain** Macro Identifier name | Compiled code **contains** Constant Identifier name |

*(Write any two of these)*

1. Write output generated by the following C++ program segment:

#define X 10.5

const Y = 10.5;

void f1()

{

const int Y = 15;

cout<<"\nX = "<<X<<", Y = "<<Y;

cout<<"\nX+Y = "<<(X + ::Y);

}

void main()

{

f1();

cout<<"\nX = "<<X<<", Y = "<<Y;

cout<<"\nX+Y = "<<(X + Y);

}

**Ans:**

X = 10.5, Y = 15

X+Y = 20.5

X = 10.5, Y = 10

X+Y = 20.5

1. Write C++ statement to generate:

i) A random number in the range -30 to -20 including -30 and -20

ii) A random uppercase alphabet

iii) A random floating point numbers in the range 10.01 to 10.99 (2 places after decimal point).

**Ans:**

**i) –(20+random(11))**

**ii) char(‘A’+random(‘Z’-‘A’+1))**

**OR**

**char(‘A’+random(26))**

**iii) (1001+random(99))/100.0**

1. Define array.

**Ans:**

Array is a **collection** of **homogeneous** data type, stored **under one variable** **name** and memory is **allocated** **contiguously**. Collection means many, homogeneous mean similar and contiguous mean a block of continuous memory location (no hole in within the block). Array name is the name of the entire block.

1. What is an array subscript (index)? What is the role of an initializer with an array?

**Ans:**

C++ compiler assigns a reference number to every element of an array. This reference number is called the array subscript (index). In C++, array subscript starts from zero (0).

1. Which of the following statements are incorrect? Support your answer with justification/reason. (No marks will be given if answers are not supported with justification/reason):

const size = 3;

int len = 5;

int A[size+2];

float B[size+len];

int C[len];

int D[] = {size, len, size+len, len-2};

char E[size] = ‘X’, ‘Y’, ‘Z’;

char F[size] = “PQRS”;

**Ans:**

|  |  |
| --- | --- |
| **Incorrect Statement** | **Reason** |
| float B[size+len]; | variable expression in array size |
| int C[len]; | variable expression in array size |
| char E[size] = ‘X’,‘Y’,‘Z’; | initialize must be in {} |
| char F[size] = “PQRS”; | initialize length more than array size |

1. What is the role of an initializer with an array?

**Ans:**

Initializer is a way to assign values to elements of an array. An initializer can only be used with an array, when the array is getting created. For example, the statement:

**int** a[5]={1, 3, 9, 6, 4};

declares array a[5] and initializes the elements to 1, 3, 9, 6, 4 in sequence.

1. Rewrite the program after removing all the syntax errors (**underline the corrections**):

#include <iostream.h>

#define MAX "10";

**void** main()

{

**int** array[MAX];

**for** (**int** x=0; x<MAX; x++)

array=random();

**for** (k=0; k<MAX; k++)

cout>>array[k];

}

**Ans:**

#include <iostream.h>

**#include <stdlib.h>**

#define MAX **10**

void main()

{

int array[MAX];

for (int x=0; x<MAX; x++)

array**[x]**=random(**20**);

for (**int** k=0; k<MAX; k++)

cout**<<**array[k];

}

1. Write output generated by the following C++ program:

#include <iostream.h>

**const** MAX=8;

**void** input(**int** a[], **int** n) { **for** (**int** k=1; k<n; k++) a[k]=5\*k; }

**void** process(**int** b[], **int** n)

{

**for** (**int** k=0; k<n; k++) b[k]=b[n-k-1];

}

**void** output(**int** c[], **int** n)

{

**for** (**int** k=0; k<n; k++)

cout<<c[k]<<"\*";

cout<<endl;

}

**void** main()

{

**int** arr[MAX]={12, 17, 15, 16, 18, 13, 14, 19};

output(arr, MAX);

input(arr, MAX);

output(arr, MAX);

process(arr, MAX);

output(arr, MAX);

}

**Ans:**

**12\*17\*15\*16\*18\*13\*14\*19\***

**12\*5\*10\*15\*20\*25\*30\*35\***

**35\*30\*25\*20\*20\*25\*30\*35\***

1. Write output generated by the C++ program below:

#include <iostream.h>

#include <ctype.h>

**void** main()

{

**char** string[]="fAxmOdeM";

**for** (**int** x=0; string[x]; x++)

**if** (islower(string[x]))

string[x]=string[x+1];

**else**

x%2 ? string[x]=tolower(string[x]) : string[x]--;

cout<<string<<endl;

}

**Ans:**

**AamONeMm**

1. Observe the following code and based on this answer the questions that follow:

**#include <iostream.h>**

**#include <conio.h>**

**const int MAX = 5;**

**void main()**

**{**

**int A[MAX];**

**for (int i = 0; i<MAX; i++)**

**A[i] = random(MAX);**

**for (int i = 0; i<MAX; i++)**

**cout<<A[i]<<”, ”;**

**}**

1. This program gives the same output on each run. Write a single statement that can make this program give different output on each run.
2. What minimum change should be made in this program so that it displays 10 random numbers?
3. What minimum change shouldbe made in the program so that random numbers are generated in the range 99 to 199?

**Ans:**

1. Write randomize(); before the first loop.
2. const int MAX = 10;
3. 99 + random(101) instead of random(MAX)
4. Define string variable.

**Ans:**

**A string variable is an array of characters terminated by a Nul character ('\0').**

1. Write a function to generate and display 10 random lowercase alphabets.

**Ans:**

**void** lowercase()

{

randomize();

**for** (**int** x=0; x<20; x++)

{

**char** ch=**char**(‘a’+random(26));

cout<<ch<<' ';

}

}

1. Write a C++ function to sort an array of floating point values (double) in descending order using Bubble sort. Array name and number of elements in the array are passed as parameters to the function. Return type of the function is void.

**Ans:**

**void** bubblesort(**double** arr[], **int** n)

{

**for** (**int** x=1; x<n; x++)

**for** (**int** k=0; k<n-x; k++)

**if** (arr[k]<arr[k+1])

{

**double** temp=arr[k];

arr[k]=arr[k+1];

arr[k+1]=temp;

}

}

1. Write a C++ function to sort an array of integers in ascending order using Selection sort. Array name and number of elements in the array are passed as parameters to the function. Return type of the function is void.

**Ans:**

**void** selectionsort(**int** arr[], **int** n)

{

**for** (**int** x=0; x<n-1; x++)

{

**int** min=arr[x], pos=x;

**for** (**int** k=x+1; k<n; k++)

**if** (arr[k]<min)

{

min=arr[k];

pos=k;

}

arr[pos]=arr[x];

arr[x]=min;

}

}

1. Write a C++ function to insert a value in an array. Array name, number of elements currently in the array, position for insertion and the value to be inserted are passed as parameters to the function. Assume that the size of the array is a user defined constant MAX. Return type of the function is void.

**Ans:**

**void** arrayins(**int** arr[], **int**& n, **int** pos, **int** item)

{

**if** (n==MAX) //Size of array is MAX

cout<<"Overflow\n";

**else**

{

**for** (**int** x=n-1; x>=pos; x--)

arr[x+1]=arr[x];

arr[pos]=item;

n++;

cout<<item<<" inserted in the array\n";

}

}

1. Write a C++ function to find the sum of all the odd integer which are stored at odd index. Array name and number of elements in the array are passed as parameters to the function. Return value of the function is int. For example if the array contains following values:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 10 | 15 | 20 | 30 | 50 | 55 | 80 | 45 | 40 | 60 |

The return value of the function should be: 15+55+45=115

**Ans:**

**void** sumodd(**int** ar[], **int** n)

{

**int** sum=0;

**for**(**int** k=0; k<n; k++)

**if** (ar[k]%2!=0)

sum+=ar[k];

return sum;

}

1. Write a C++ function to count and display number of uppercase characters, number of lowercase characters, number of digits and number of special character present in a string passed as a parameter to the function. Return type of the function is void. For example, if the string contains:

"Tower-B/106, Ganga Apartment, Kundan Vihar, NEW DELHI-110028"

Then the output should be:

Uppercase=14

Lowercase=25

Digit=9

Special Characters=12

**Ans:**

**void** countchars(**char** str[])

{

**int** u=0, l=0, d=0, s=0

**for** (**int** x=0; str[x]; x++)

**if** (str[x]>='A' && str[x]<='Z')

u++;

**else if** (str[x]>='a' && str[x]<='z')

l++;

**else if** (str[x]>='0' && str[x]<='9')

d++;

**else**

s++;

cout<<"Uppercase="<<u<<endl<<"Lowercase="<<l<<endl;

cout<<"Digits="<<u<<endl<<"Special Characters="<<s<<endl;

}

1. Write a C++ function to check whether a string which is passed as a parameter to the function is a Palindrome or not. Return type of the function is void. Do not use any function from the header files <ctype.h> and <string.h>.

**Ans:**

**void** chkpalindrome(**char** str[])

{

**int** len=0, palin=1

**while** (str[len])

len++;

**for** (**int** x=0; x<len/2 && palin==1; x++)

**if** (str[x]!=str[len-x-1])

palin=0;

**if** (palin == 1)

cout<<”Palindrome”;

**else** cout<<”Not Palindrome”;

}

1. Write a C++ function to toggle the characters of a string passed as a parameter to the function. Display the toggled string on the screen. Return type of the function is void. For example, if the string contains:

gOoD LuCK

Then the output string should be: GoOd lUck.

**Ans:**

**void** stringtoggle(**char** str[])

{

**for**(**int** x=0; str[x]; x++)

**if** (str[x]>='A' && str[x]<='Z')

str[x]=**char**(str[x]+32);

**else**

**if** (str[x]>='a' && str[x]<='z')

str[x]=**char**(str[x]-32);

}