Math is fun.com is game web site for maths students

**C++:**

# include means we are importing libraries(header files)

<library name>

cout<<””; use to print data

puts(“”); also use to print data

cin>>var use to take input from user, input will save in variable

() called as paranthesis

[] called as brackets/square brackets

{} calles as braces / curley braces

Method to Take data from user: getline(waytotakedata,variableToStoreData) like getline(cin,number)

Using namespace std:

Std mean standard, using namespace std means every line of code after this will be coming from namespace standard file.

It is a way to group named entities(like variable, function) under single scope.

%d is known as place holder it takes the value.

Like printf(“the calculated value is %d,2+6);

Output: the calculated value is 8

**CPP Identifiers:**

cpp is case sensitive.

If upper and lower case then separate memory locations will be allocated.

Numbers should be at the end or between of variable name, it should not be in start.

\_ can be used to start a variable.\_ mean private character, two \_ at start means reserved keywords

Cannot used reserved keywords(75)

Non-latin(urdu) can be used but avoid to use it.

Use maximum 37 characters I keyword.

**Character literals:**

**\n:** new line

**\\:**  \ character

**\’:** ‘character

**\”:** “character

**\?:** ? character

**\a:** alert or bell

**\b:** backspace

**\f:** form feed

**\r:** carriage return

**\t:** horizontal tab

**\v:** vertical tab

**\ooo:** octal number one to three digits

**\xhhh:** hexadecimal number of one or more dogits

**Primitive Data types:**

Bool,

char,

int,

float, => double => long double => it shows more numbers after decimal according to float<double<long double

double(it shows float value but many numbers after point),

void(null or nothing)

**non-primitive data types:**

**string**: import string before using bcz it is non-primitive e.g. #include <string>

**note:** declaration means just make variable, initializing means assigning value to the variable.

Time: 1:17:50

**Pointer:**

Pointers points to a memory location, declate with datatype then use “\*” and assign name to pointer.

e.g. int \*khanopointer;

then we have to point to some variable

e.g,

int a=50;

int \*khanopointer;

. khanopointer=&a; & amperson, it points to memory address to a, so khanopointer wll point

a’s memory address;

if we want to print memory address then we simply print . khanopointer; if we want to data then we use \*khanopointer;

like printf(\*khanopointer)

if want to use specifier then we will use %p

we can assign references and assign variable to it;

after then when we assign a new value to the reference that we decled it will change the value of the variable

e.g.

**int** a = 50;

**int** &myref =a;

myref=70000;

cout<<"the value printed from reference is"<<myref;

output will be 70000 instead of 50;

**Arrays:**

It is contigeous memory location which store multiple values of same data types.

**int** myArr[6]={5,3,2,3,4,5};

cout<<myArr[0];

it will print 5

if we just print myArr; it will print just memory address;

we can change the 1st index value by using pointer,

like \*myArr=5; it will replace 1st value with 5;

if we want to access and change the next index value then we simply create new pointer, point it to our array,

and do increment to it and assign a new value, it will change the value of index according to increment.

e.g.

**int** myArr[6]={5,3,2,3,4,5};

**int** \*arrRef=myArr;

arrRef++;

\*arrRef=80;

cout<<myArr[1]<<"\n";

arrRef++;

\*arrRef=100;

cout<<myArr[2]<<"\n";

output: 80,100

**Integers: Note:**we can check size of any data type by sung size(); method, give value in byte

The type of integers holds different data size.

1 byte=8 bits

The size of bits depends on the machine and os in which it is being used we the short int is the smallest one and the long is longest one.

Short int< int<long int< long long int

All of them store non-decimal math value

**short int: 16bits**

**int: 32 bits**

if we assign vakue to int by 0b00101110, it will convert binary code into number, before binary code we have to write ob

**long int: 64 bits**

we can simple use just long

**long long int: 64bits or more according to OS**

we can simple use just long long

if we want to fix bit size for all systems then we can use library <cstdint> to specify bits. We can use it like int16\_t a=15; now int is of maximum 16 bits,(note: if are just declaring and not going to assign value during declaration then we can use uint16\_t ) it will use at the place of long long,short, long etc

**Conditionals:**

**If-else:**

      int a= 7;

       if (a>=18)

       {

       cout<<"congratulation! your are eligible for marriage";

       }else{

        int remaining\_year=18-a;

        cout<<"sorry! you have to wait "<<remaining\_year<<" years for marriage";

       }

**Ternary operators:**

int age = 17;

    age >= 18 ? cout << "you are eligible for marriage" : cout << "sorry you have to wait for " << 18 - age << " years for marriage";

**Switch case:**

int rating = 2;

    switch (rating)

    {

    case 1:

        cout << "so bad to know you don't like our product" << "";

        break;

    case 2:

        cout << "so bad to know you don't like our product" << "";

        break;

    case 3:

        cout << "your feedback will be good to improve our product" << "";

        break;

    case 4:

        cout << "good to know you like our product we will be improve our product according to your feedback" << "";

        break;

    default:

        cout << "Good hogya, done hogya te chass agae" << "";

        break;

    }

**Loops:**

**Forloop:**

string str="kamboh";

for (int i = 0; i < str.length(); i++)

{

   std::cout << str[i] << std::endl;

}

**While loop:**

    int i=str.length();

    while (i!=-1)

    {

      cout << str[i]<<"\n";

       i--;

    }

**Dowhile loop:**

int i=9;

do

{

if (i>=str.length())

{

std::cout << "number is bigger than string length" << std::endl;

}else{

std::cout << str[i] <<std::endl;

}

i++;

} while (i<str.length());

**For Range:**

it act as forEach loop in js

works with arrays and char data types

for\_range loop: it act like for each loop

**int** arr[]={2,3,4,5};

for (**int** i : arr)

{

    std::cout << i\*2 << std::endl;

}

**Loop with pointers:**

  char ch[]="ali\_akbar";

for (char \*mypntr = ch; \*mypntr!=0; mypntr++)

{

*/\* code \*/*

    std::cout << \*mypntr << std::endl;

}

**Try\_Catch:**

When we are writing some that that may give some error like Api calling then it may crash the code so we handle the error by using try catch so other code will not effect by this error

in catch block we have to define datatype which we are going to accept. Like catch(int err){}

but if we don’nt know which type of error will come then we can use … like catch(…){}

in try block we through error of type that may cause error, after through the rest of the other code will not be executed so so should through at the end of block

**Example:**

**float** num = 5;

    try

    {

        cout << "1st code in try block\n";

        cout << "2nd code in try block\n";

        throw num;

        cout << "the code after throw error\n";

    }

    catch (**int** err)

    {

        std::cout << "the error cause by this num which should give value: " << num << std::endl;

    }

    catch (...)

    {

        std::cout << "all type of data error can be executed through this block" << std::endl;

    }

**Functions:**

Before start the function we have to define type which the function is going to return.

We have to also define the type of argument that we are going to receive.

If a function is defined with a datatype then it is necessary that It have to return that data type

If we have multiple function then all of the function should be called inside the main function

**Linker:**

When we import library, before compile the import library file with our code file compile together and make an object, the object goes to linker, linker take standard runtime library to make .exe executable file.

Object is a numeric binary code that our coputer architecture can only understand

**Qualifiers:**

Two types:

1. Modification qualifier

Allow to modify or not a value, like const

Types:

* Const: if value assigned with cost then we cannot modify it later
* Volatile: use to share one variable from one thread to another thread
* Mutable: it make variables mutable

1. Life duration qualifier

How much time or area variable is available to me

Types:

* Static:
* Register: give suggestion to compiler to store information to register(use while hardware handeling)
* Extern: use to store something in outerlibrary which will be done at the time linker