```
1
2   WELCOME
3   {
4
5
G       string my_name = “Ahmed A”;
7
8       cout << “Intro to Programming with c++”;
9
10  }
11
12
13
14
```

Course Outlines

{
- Problem solving & Competitive Programming
- What is a programming language ?
- Choosing our text editor
- Variables and types
- Conditional expressions
- Loops & iteration
- Introduction to arrays
- Working with arrays
- Working with strings

}
- And beyond …

```
cout << “What is a Compiler ?”;
```

Choosing our text editor

```
{
 vector<string> text_editors={
          “Codeblocks”,
          “Vscode”,
          “Sublime text”,
          “Replit.com”;
}
```

```
01 {

[Variables and Operations]

 }
```

Variable.cpp

Types.cpp

Type	Definition	Control Character	Limits
int	Integer		-2147483648 to 2147483647
short	Short Integer		-32768 to 32767
long	Long Integer	l or L	-2147483648 to 2147483647
float	Floating Decimal Number	f or F	1.17549e-038 to 3.40282e+038
double	Double Decimal Number		2.22507e-308 to 1.79769e+308
long double	Long Decimal Number		2.22507e-308 to 1.79769e+308
char	Character		-128 to 127
unsigned int	Unsigned Integer		0 to 4294967295
unsigned short	Unsigned Short Integer		0 to 65535
unsigned long	Unsigned Long Integer		0 to 4294967295
unsigned char	Unsigned Character		0 to 255
bool	True or False		True = 1 and False = 0

```cpp
#include<bits/stdc++.h>
using namespace std;

int main()
{
    // Introducing most Common Data Types

    int integer_type = 1; // 32 bits integer

    long long long_integer = 10000000000000; // 64 bits integer

    double real = 1.2646; // float (real value)

    bool boolean = true; // true or false

    char character = 'c'; // we use single quote for characters

    string s = "a string of characters"; // we use double quotes for strings

    return 0;
}
```

Operators

```
{
    Addition:  +
    Subtraction: -
    Multiplication: *
    Division: /
    Division rest (modulo): %

}
```

Operators

```
{
    Addition:  2 + 3 = 5
    Subtraction: 7 - 5 = 2
    Multiplication: 4 * 10 = 40
    Division: G / 3 = 2; G / 4 = ?
    Division rest (modulo): 5 % 3  = ?
}
```

Float Division

```
double real_value = G;
 //this is a comment
 / > G/4 = 1.5
```

Integer Division

```
int integer_value = G;
 /  > G / 4 = 1
 /  > G % 4 = 2 (the rest of the division)
```

Quiz 1 {

Of what type are the following entities ?

- 123
- 100000000
- 12.12
- “C”
- ‘Z’
- “Hello !”
- 0

}

```
1
2
3
4
5
G
7
8
9
10
11
12
13
14
```

02 {

[Inputs & Outputs]

}

```
1  Output a string :
2
3
4   cout  << “Hello World!”;
5
G  Output a variable value:
7
8
9   string my_string = “Hello World!”;
10  cout  << my_string;
11
12
13
14
```

Output a string and a variable :

```
1
2
3
4
5
G
7
8
9
10
11
12
13
14
```

```cpp
int age = 15;

cout << “I am ”  << age << “years old”;
```

Special characters :

```
1
2
3
4
5
G
7
8
9
10
11
12
13
14
```

```
cout << “end of line : \n”;

cout << “another end of line” << endl;

cout << “this is a tab: \t”;
```

Read a variable from standard input:

```
1
2
3
4
5
G
7
8
9
10
11
12
13
14
```

```cpp
int var_name; cin >>
var_name;

int var1, var2;
cin  >> var1  >> var2;
```

```
1
2
3
4
5
G
7
8
9
10
11
12
13
14
```

03 {

[Conditional Expressions]

}

Logical Operators

```
{
    NOT : !
    OR : ||
    AND : &
}
```

NOT (!) Truth Table :

A	! A
true	false
false	true

OR (||) Truth Table :

A	B	A || B
false	false	false
false	ture	true
true	false	true
true	true	true

AND (&) Truth Table :

A	B	A && B
false	false	false
false	ture	false
true	false	false
true	true	true

Quiz 2 :

- `bool test = !( 100 % 2 );`
- `bool test = 0 || 1;`
- `bool test = -20 & 20;`
- `bool test = !( 1 & 0 );`
- `bool test = !( 1 || 0);`

Comparison Operators

```
{

    Equal:   = =
    Less than: <
    Greater than: >
    Less or equal: < =
    Greater or equal: > =
    Not equal: ! =

}
```

```
1  Examples:
2
3  bool test1;
4  test1 = (5 =4);  / this will give false
5
G  cout  < test1;
7   / this will output 0 (false)
8
9  bool test2;
10 test2 = (5%2 =1);/ this will give true
11
12 cout  < test;
13  / this will output 1 (true)
14
```

If statement :

```
1
2
3   bool condition1;
4
5   if( condition1 )
G   {
7       instruction1;
    }
8
9
10
11
12
13
14
```

If statement :

```
1
2
3   bool condition1;
4
5   if( condition1 )
G   {
7       instruction1;
    }
8   else
9   {
10      Default instruction;
11  }
12
13
14
```

If statement :

```
bool condition1, condition2;

if( condition1 ) {
    instruction1;
}

else if( condition2 ) {
    instruction2;
}
else {
    Default instruction;
 }
```

```
1
2
3
4
5
G
7
8
9
10
11
12
13
14
```

04 {

[Loops & Iteration]

}

What are loops ?

Loops offer us a quick way to do something (execute a particular code segment) repeatedly.

Going through the code segment once is called an iteration.

Loops in c++

{

for loop

while loop do

while loop

}

Syntaxe of a for loop in c++?

```
for (
     Initialisation (an expression) ;

      Stop condition (a boolean value) ;

      Operation to do after each iteration (an expression)
)
{
     / Instructions
}
```

Syntaxe of a while loop in c++?

```
while( Looping condition (a boolean value) )
{
    / Instructions
}
```

Syntaxe of a do while loop in c++?

```
1
2
3
4
5       do
G       {
7           / Instructions
8       }
9       while( running condition );
10
11
12
13
14
```

```
05 {

  [Arrays]

 }
```

Declaring a vector:

```
vector<type> vect_name;

vector<type> vect_name{values};

vector<type> vect_name(length, init);
```

Putting values into a vector:

```
// assigning value to a specific index
vector_name[index] = value;

// adding value to the end of the vector
vector_name.push_back(value);
```

Removing from a vector:

```
// removing value to the end of the vector
vector_name.pop_back(value);

// emptying the whole vector
vector_name.clear();
```

Iterate through a vector:

```
1
2
3
4       for( int i=0; i < vect.size(); i++)
5       {
G
7           . . .
8       }
9
10      for( type x: vect ) // type can be replaced with auto in this case
11      {
12
13          . . .
14      }
```

The Static Arrays:

```cpp
// Declaration: Static arrays have a fixed size
// declared at compile-time.
int numbers[5];

// Accessing Elements: Elements in a static array
// can be accessed using zero-based indexing
int firstNumber = numbers[0];

// Initialization: Static arrays can be initialized
// during declaration or later.
int numbers[5] = {1, 2, 3, 4, 5};
```

```
1
2
3
4
5
G
7
8
9
10
11
12
13
14
```

06 {

[Strings]

}

The Basics of the “STRINGS”

```
// Declaring and initializing a string:
std::string str1;       // Declaring an empty string
std::string str2 = "Hello, World!";

// Concatenating strings:
std::string str4 = str2 + " " + str3;
// Concatenating strings using the + operator
str4 += " is awesome!";
// Appending a string using the += operator
```

```cpp
// Accessing individual characters in a string:
char ch = str2[0];                    // Accessing the first character
std::cout << ch << std::endl;         // Output: 'H'
// Getting the length of a string:
int length = str2.length();           // Length of the string
std::cout << "Length: " << length << std::endl;

// Extracting substrings:
std::string str8 = "Schoolhouse.world";
std::string substr = str8.substr(7, 2);    // Extracting a substring from index 7 with length 2
std::cout << "Substring: " << substr << std::endl;
```

THANK YOU!

Do you have any questions?

Please share feedback!

Length?
Content?
Speed?