GIANI ZAIL SINGH CAMPUS COLLEGE
OF
ENGINEERING & TECHNOLOGY
BATHINDA

TRAINING REPORT

On

PROGRAMMING WITH C++

Submitted to MAHARAJA RANJIT SINGH PUNJAB TECHNICAL UNIVERSITY in partial fulfillment of the requirement for the award of the degree of

B.TECH

in

COMPUTER SCIENCE & ENGINEERING

Submitted By
KASHIF QUAMAR
(Roll.No.190280063)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

GIANI ZAIL SINGH CAMPUS COLLEGE OF ENGINEERING& TECHNOLOGY, MRSPTU, BATHINDA-151001

JUNE-AUGUST 2021

TRAINING REPORT

On

PROGRAMMING WITH C++

Submitted to MAHARAJA RANJIT SINGH PUNJAB TECHNICAL UNIVERSITY in partial fulfillment of the requirement for the award of the degree of

B.TECH

in

COMPUTER SCIENCE & ENGINEERING

Submitted By
KASHIF QUAMAR
Roll.No.190280063

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

GIANI ZAIL SINGH CAMPUS COLLEGE OF ENGINEERING & TECHNOLOGY, MRSPTU, BATHINDA-151001

JUNE-AUGUST 2021

SwarnimaShree INFOTECH Pvt. Ltd.

SOFTWARE DEVELOPMENT & TRAINING CENTER

CIN : U72900BR2018PTC4026

Certificate NO. : **SSI-3808456765**

Enrollment No. : **SSIBR5748811715**

Certificate

This is to certify that

KASHIF QUAMAR

D/o S/o W/o

MD SHABBIR ALAM

Course

C++ PROGRAMMING LANGUAGE

in the period of

10/06/2021 TO 27/07/2021

He/She has completed the above mentioned course with

Grade A+

Evaluator

Managing Director

Date of Issue : **30th JUL. 2021**

Head Office : Shiv Appt., Opp. Gayatri Mandir, Kankarbagh, Patna-800 020
Email : infotechpatna@gmail.com
Website : www.swarnimashreeinfotech.com

CERTIFIED ISO 9001:2015 COMPANY

ISO Certificate No.
19DQEF55

PREFACE

Training is an integral part of B.Tech and each and every student has to undergo the training for **7 weeks** in a company.

This record is concerned about our practical training during the **5th** semester of our B.Tech. I have taken our Practical training in **SWARNIMASHREE INFOTECH Pvt Ltd** During this training, I got to learn many new things about the industry and the current requirements of companies. This training proved to be a milestone in our knowledge of present industry. Every say and every moment was an experience in itself, an experience which theoretical study can't provide.

ACKNOWLEDGEMENT

It is my pleasure to be indebted to various people, who directly or indirectly contributed in the development of this work and who influenced my thinking, behavior and acts during the course of study.

I express my sincere gratitude to ***Dr. JYOTI RANI*** worthy HOD and ***Er. Naresh Garg and Er. Manpreet Kaur***, Training & Placement In-charge for providing me an opportunity to undergo summer training at **SWARNIMASHREE INFOTECH Pvt Ltd.**

I am thankful to **ATUL RANJAN SIR** for his support, cooperation, and motivation provided to me during the training for constant inspiration, presence and blessings.

I also extend my sincere appreciation to ***my colleagues*** who provided his valuable suggestions and precious time in accomplishing my training report.

Lastly, I would like to thank the almighty and my parents for their moral support and my friends with whom I shared my day-to-day experience and received lots of suggestions that my quality of work.

KASHIF QUAMAR

(190280063)

CANDIDATE'S DECLARATION

I, **KASHIF QUAMAR**, Roll No**. 190280063** , B.Tech (Semester- V) of the **Gaini Zail Singh Campus College of Engineering & Technology, Bathinda** hereby declare that the Training Report entitled **"C++ PROGRAMMING IN SWARNIMASHREE INFOTECH Pvt Ltd"** is an original work and data provided in the study is authentic to the best of my knowledge. This report has not been submitted to any other Institute for the award of any other degree.

KASHIF QUAMAR
(Roll No.190280063**)**

PLACE: GZSCCET,MRSPTU,BATHINDA
DATE : 27/01/2022

STUDENT PROFILE

Particulars of Student and Training		Remarks
Name of the student	KASHIF QUAMAR	
University Roll No.	190280063	
Phone No.	7484074531	
e-mail ID	kashifqumar06@gmail.com	
Class	B. Tech. Computer Science and Engg.	
Semester & Session	5th / July – Dec 2021	
Subject Name	Institutional/Industrial Training (6-week)	
Subject Code	BCSES1-5O7	
Credits	4	
Max. Internal Marks	60	
Max. External Marks	40	
Name of Industry	SwarnimaShree Infotech Pvt. Ltd ,Patna	
Training Period	6-Weeks	
Training Dates From – To	10/06/2021 To 27/07/2021	
Declaration by the student	It is certified that this Training Report is of my own work.	
Date of Internal Presentation & Viva-Voce	28/01/2022	

Name & Sign.
Internal Examiner

Name & Sign.
Internal Examiner

External Examination of the candidate has been held on date _ _ _ _ _ _ _ _ _ _ and found to be _ _ _ _ _ _ _ _ _ _ _ _.

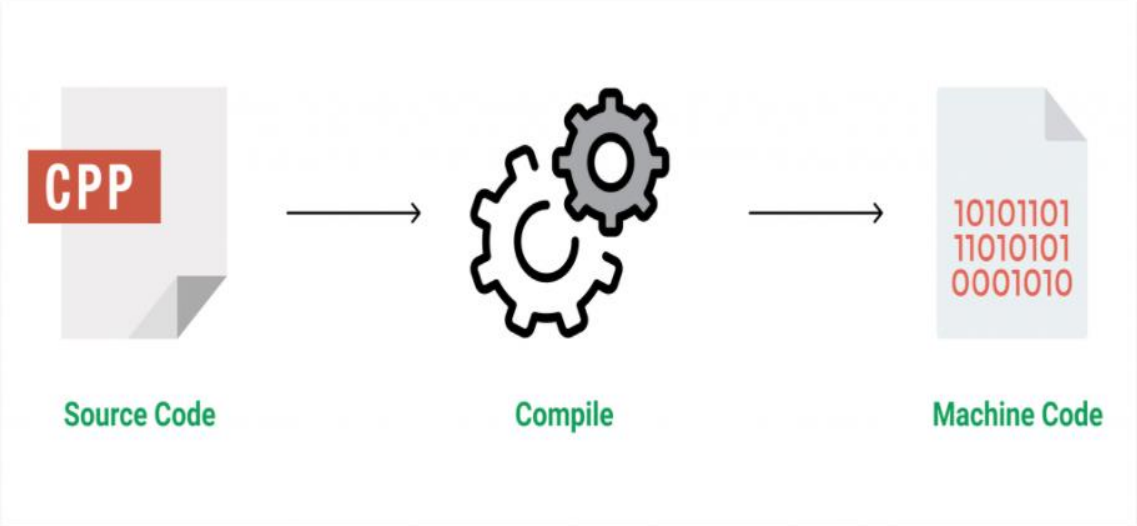
Name & Sign.
Internal Examiner

Name & Sign.
External Examiner

Name & Sign.
Head of EE Department

INTRODUCTION

C++ is a cross-platform language that cn be used to create high-performance applications.C++ was deleloped by Bjarne stroustrup, as an extension to the c language.C++ gives programmers a high level of control over system resources and memory.The language has expanded significantlyover time, and modern C++ now has object-oriented, generic, and functional features in additionto facilities for low-level memory manipulation. It is almost always implemented as a compiledlanguage, and many vendors provide C++ compilers, including the Free Software Foundation,LLVM, Microsoft, Intel, Oracle, and IBM, so it is available on many platforms.C++ was designed with a bias toward system programming and embedded, resource-constrainedsoftware and large systems, with performance, efficiency, and flexibility of use as its



Sample program

```
#include <iostream>
    using namespace std;
      int main() {
      cout << "Hello World!";
```

Why use C++

C++ is one of the world's most popular programming languages.
C++ can be found in today's opearting systems,Graphical user interface,and embedded systems.C++ is an object-oriented programming language which gives a clear structure to programs and allows code to be reused,lowering development costs.
C++ is portable and can be used to delelop applications that can be adapted to multiple platforms.

Application of C++

- **Application Software Development** - C++ programming has been used in developing almost all the major Operating Systems like Windows, Mac OSX and Linux. Apart from the operating systems, the core part of many browsers like Mozilla Firefox and Chrome have been written using C++. C++ also has been used in developing the most popular database system called MySQL.
- **Programming Languages Development** - C++ has been used extensively in developing new programming languages like C#, Java, JavaScript, Perl, UNIX's C Shell, PHP and Python, and Verilog etc.
- **Computation Programming** - C++ is the best friends of scientists because of fast speed and computational efficiencies.
- **Games Development** - C++ is extremely fast which allows programmers to do procedural programming for CPU intensive functions and provides greater control over hardware, because of which it has been widely used in development of gaming engines.
- **Embedded System** - C++ is being heavily used in developing Medical and Engineering Applications like softwares for MRI machines, high-end CAD/CAM systems etc.

❖ CHAPTER NO-1

1.1 VARIABLES

- A variable is a name of memory location. It is used to store data. Its value can be changed and it can be reused many times.
- It is a way to represent memory location through symbol so that it can be easily identified.

1.2 DATATYPES

- A data type specifies the type of data that a variable can store such as integer, floating, character etc.
- There are mainly divided into two parts
  1) Primitive data type
  2) Derived data type

1) Primitive data types-these data types are built in or predefined dat types and can be used directly by the user to declare variables like
   INTEGER

CHARACTER
BOOLEAN
FLOATING POINT
DOUBLE FLOATING POINT
VALUELESS OR VOID
WIDE CHARACTER

2)Derived data types=the data types that are derived from primitive are referred as derived data type.Thsese can be of four types namely:

- Function
- Array
- Pointer
- Reference

3) Abstract or user defined data types:these data types are defined by user itself. Following are the user defined data types
   - Class
   - Structure
   - Union
   - Enumeration
   - Typedef defined datatype

1.3 Keywords-

A keyword is a reserved word. You cannot use it as a variable name, constant name etc**.** A list of 32 Keywords in C++ Language which are also available in C language are given below.

Uto	break	case	char	const	continue	default	do
double	else	enum	extern	float	for		goto
Int	long	register	return	short	signed		sizeof
struct	switch	typedef	union	unsigned	void		volatile

1.4 Control statement

A c++ control statement redirect the flow of a program in order to execute the additional code.

In C++ programming, if statement is used to test the condition. There are various types of if statements in C++.

- if statement
- if-else statement
- nested if statement
- if-else-if ladder

1.4.1 if statement

The C++ if statement tests the condition. It is executed if condition is true.

```
1. if(condition){
2. //code to be executed
3. }
```

If statement example:-

```
1. #include <iostream>
2. using namespace std;
3.
4. int main () {
5.   int num = 10;
6.         if (num % 2 == 0)
7.         {
8.            cout<<"It is even number";
9.         }
10.  return 0;
11. }
```

1.4.2 if-else statement:-

The C++ if-else statement also tests the condition. It executes if block if condition is true otherwise else block is executed.

```
1. if(condition){
2. //code if condition is true
3. }else{
4. //code if condition is false
5. }
```

If-else example:-

```
1. #include <iostream>
2. using namespace std;
3. int main () {
4.   int num = 11;
5.         if (num % 2 == 0)
6.         {
7.            cout<<"It is even number";
```

```
8.          }
9.          else
10.         {
11.             cout<<"It is odd number";
12.         }
13.   return 0;
14. }
```

1.4.3 nested if else

When a series of decision is required, nested if-else is used. Nesting means using one if-else construct within another one.

Example:-

```
#include<stdio.h>
int main()
{
        int num=1;
        if(num<10)
        {
                if(num==1)
                {
                        printf("The value is:%d\n",num);
                }
                else
                {
                        printf("The value is greater than 1");
                }
        }
        else
        {
                printf("The value is greater than 10");
        }
        return 0;
}
```

1.5 Loops in c++

Loops can execute a block of code as long as a specified condition is reached. Loops are handy because they save time, reduce errors, and they make code more readable,

1.5.1 while loop

In C++, while loop is used to iterate a part of the program several times. If the number of iteration is not fixed, it is recommended to use while loop than for loop.

Synyax:-

```
1. while(condition){
2. //code to be executed
3. }
```

Example:-

```
1. #include <iostream>
2. using namespace std;
3. int main() {
4.  int i=1;
5.          while(i<=10)
6.        {
7.            cout<<i <<"\n";
8.            i++;
9.          }
10.    }
```

1.5.2 do-while loop

The C++ do-while loop is used to iterate a part of the program several times. If the number of iteration is not fixed and you must have to execute the loop at least once, it is recommended to use do-while loop.

Syntax-

```
1.      do {
2. //code to be executed
3. }while(condition);
```

Example-

```
1. #include <iostream>
2. using namespace std;
3. int main() {
4.     int i = 1;
5.          do{
6.             cout<<i<<"\n";
7.             i++;
8.          } while (i <= 10) ;
9. }
```

1.5.3 for-loop

The C++ for loop is used to iterate a part of the program several times. If the number of iteration is fixed, it is recommended to use for loop than while or do-while loops.

Syxtax-

```
for(initialization; condition; incr/decr){
//code to be executed
}
```

Example-

```
#include <iostream>
using namespace std;
int main() {
        for(int i=1;i<=10;i++){
          cout<<i <<"\n";
         }
    }
```

1.6 Arrays

Arrays are used to store multiple values in a single variable, instead of declaring separate variables for each value.

Example-

```
#include <iostream>
using namespace std;
int main()
{
 int arr[5]={10, 0, 20, 0, 30};  //creating and initializing array
       //traversing array
       for (int i = 0; i < 5; i++)
       {
          cout<<arr[i]<<"\n";
       }
}
```

1.7 Functions

A function is a group of statements that together perform a task. Every C++ program has at least one function, which is **main()**, and all the most trivial programs can define additional functions.

Syntax-

```
return_type function_name( parameter list ) {
body of the function
}
```

Example-

```
// function returning the max between two numbers

  int max(int num1, int num2) {
 // local variable declaration
   int result;
 if (num1 > num2)
    result = num1;
else
   result = num2;

  return result;
 }
```

1.8 Object oriented programming

OOP stands for Object-Oriented Programming.

Procedural programming is about writing procedures or functions that perform operations on the data, while object-oriented programming is about creating objects that contain both data and functions.

Object-oriented programming has several advantages over procedural programming:

OOP is faster and easier to execute

OOP provides a clear structure for the programs

OOP helps to keep the C++ code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug

OOP makes it possible to create full reusable applications with less code and shorter development time

1.9 classes and object

Objects are the basic run-time entities in an object-oriented system. They may represent a person, a place, a bank account, a table of data or any item that the program must handle.

C++ is an object-oriented programming language.

Everything in C++ is associated with classes and objects, along with its attributes and methods. For example: in real life, a car is an object. The car has attributes, such as weight and color, and methods, such as drive and brake.

Attributes and methods are basically variables and functions that belongs to the class. These

are often referred to as "class members".

A **class** is a user-defined data type that we can use in our program, and it works as an object constructor, or a "blueprint" for creating objects.

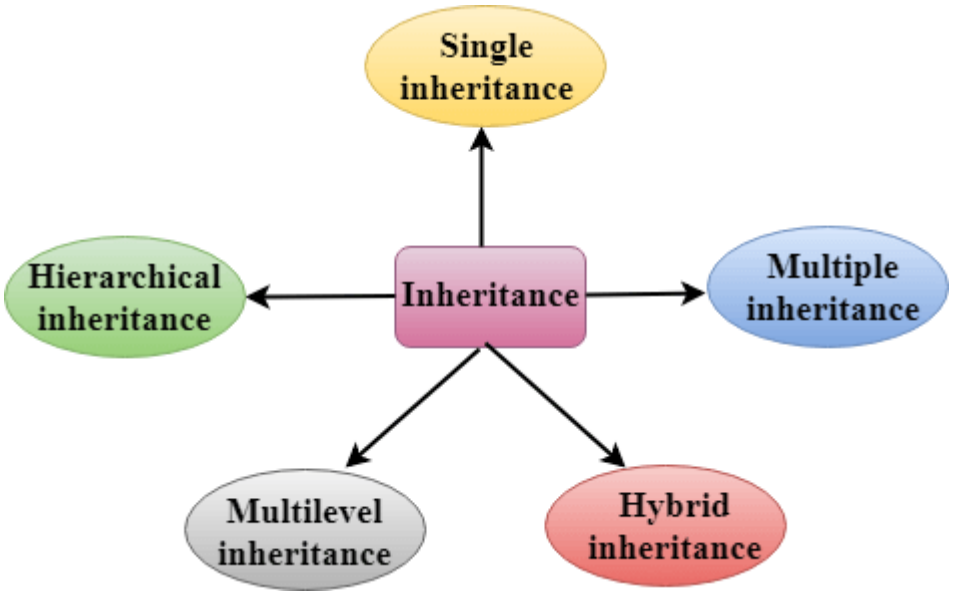
1.10 Inheritance in C++

In C++, inheritance is a process in which one object acquires all the properties and behaviors of its parent object automatically. In such way, you can reuse, extend or modify the attributes and behaviors which are defined in other class.

In C++, the class which inherits the members of another class is called **derived class** and the class whose members are inherited is called **base class**. The derived class is the specialized class for the base class.

C++ supports five types of inheritance:

- Single inheritance
- Multiple inheritance
- Hierarchical inheritance
- Multilevel inheritance
- Hybrid inheritance



Example-

```
1.  #include <iostream>
2.  using namespace std;
3.  class A
4.  {
```

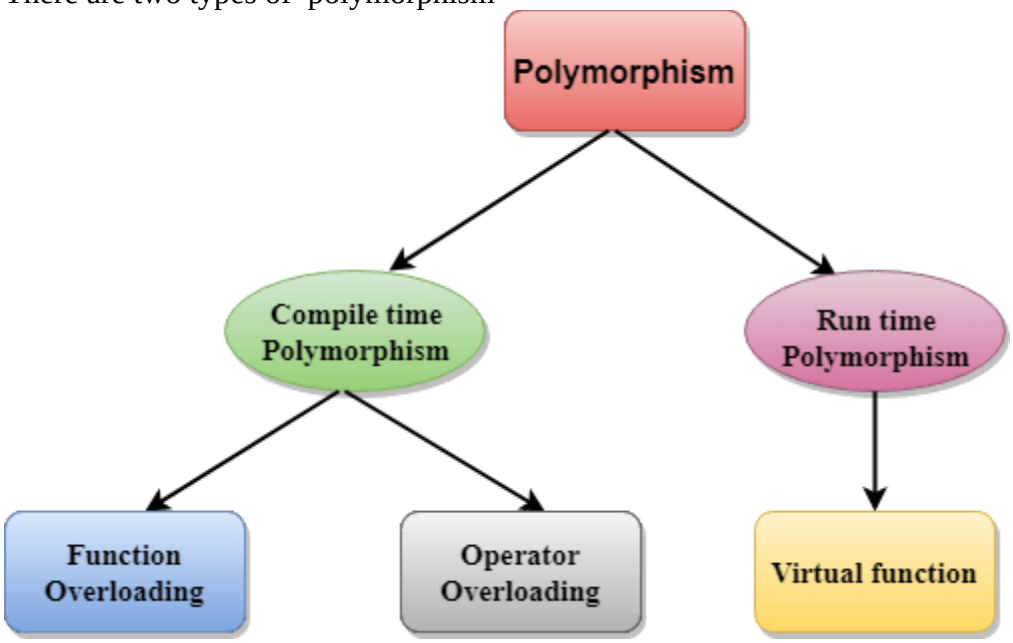
```
5.      protected:
6.      int a;
7.      public:
8.      void get_a()
9.      {
10.       std::cout << "Enter the value of 'a' : " << std::endl;
11.       cin>>a;
12.     }
13. };
14.
15. class B : public A
16. {
17.     protected:
18.     int b;
19.     public:
20.     void get_b()
21.     {
22.        std::cout << "Enter the value of 'b' : " << std::endl;
23.       cin>>b;
24.     }
25. };
26. class C
27. {
28.     protected:
29.     int c;
30.     public:
31.     void get_c()
32.     {
33.        std::cout << "Enter the value of c is : " << std::endl;
34.        cin>>c;
35.     }
36. };
37.
38. class D : public B, public C
39. {
40.     protected:
41.     int d;
```

```
42.     public:
43.     void mul()
44.     {
45.         get_a();
46.         get_b();
47.         get_c();
48.         std::cout << "Multiplication of a,b,c is : " <<a*b*c<< std::endl;
49.     }
50. };
51. int main()
52. {
53.     D d;
54.     d.mul();
55.     return 0;
    }
```

1.11 Polymorphism

The term "Polymorphism" is the combination of "poly" + "morphs" which means many forms. It is a greek word. In object-oriented programming, we use 3 main concepts: inheritance, encapsulation, and polymorphism.

There are two types of polymorphism



Compile time polymorphism: The overloaded functions are invoked by matching the type and number of arguments. This information is available at the compile time and, therefore, compiler selects the appropriate function at the compile time. It is achieved by function overloading and operator overloading which is also known as static binding or early binding.

Example-

```
class A                          //  base class declaration.
 {
    int a;
    public:
    void display()
    {
        cout<< "Class A ";
     }
 };
class B : public A               //  derived class declaration.
{
  int b;
  public:
 void display()
 {
    cout<<"Class B";
 }
};
```

- **Run time polymorphism**: Run time polymorphism is achieved when the object's method is invoked at the run time instead of compile time. It is achieved by method overriding which is also known as dynamic binding or late binding.

Example-

```
#include <iostream>
using namespace std;
class Animal {
  public:
void eat(){
cout<<"Eating...";
```

```
7.      }
8.  };
9.  class Dog: public Animal
10. {
11.  public:
12.  void eat()
13.     {          cout<<"Eating bread...";
14.     }
15. };
16. int main(void) {
17.    Dog d = Dog();
18.    d.eat();
19.    return 0;
20. }
```

1.12 Abstraction in C++

Abstract classes are the way to achieve abstraction in C++. Abstraction in C++ is the process to hide the internal details and showing functionality only. Abstraction can be achieved by two ways:

1. **Abstract class**
2. **Interfaces**

1)Abstract class-

In C++ class is made abstract by declaring at least one of its functions as pure virtual function. A pure virtual function is specified by placing "= 0" in its declaration. Its implementation must be provided by derived classes.

Example-

```
1.  #include <iostream>
2.  using namespace std;
3.   class Shape
4.  {
5.     public:
6.     virtual void draw()=0;
7.  };
```

```
8.  class Rectangle : Shape
9.  {
10.    public:
11.     void draw()
12.    {
13.       cout < <"drawing rectangle..." < <endl;
14.    }
15. };
16. class Circle : Shape
17. {
18.    public:
19.     void draw()
20.    {
21.       cout <<"drawing circle..." < <endl;
22.    }
23. };
24. int main( ) {
25.    Rectangle rec;
26.    Circle cir;
27.    rec.draw();
28.    cir.draw();
29.   return 0;
30. }
```

1.13 Encapsulation

Encapsulation is an Object Oriented Programming concept that binds together the data and functions that manipulate the data, and that keeps both safe from outside interference and misuse. Data encapsulation led to the important OOP concept of data hiding.

Data encapsulation is a mechanism of bundling the data, and the functions that use them and data abstraction is a mechanism of exposing only the interfaces and hiding the implementation details from the user.

Example-

```
class A                          //  base class declaration.
{
    int a;
```

```
    public:
   void display()
     {
          cout<< "Class A ";
     }   };
class B : protected A                    //  derived class declaration.
{
   int b;
   public:
  void display()
 {
      cout<<"Class B";
 }
};
```

1.14 Constructor in c++

In C++, constructor is a special method which is invoked automatically at the time of object creation. It is used to initialize the data members of new object generally. The constructor in C++ has the same name as class or structure.

There can be two types of constructors in C++.

- Default constructor
- Parameterized constructor

Example-

```
#include <iostream>
using namespace std;
class Employee
 {
  public:
     Employee()
     {
        cout<<"Default Constructor Invoked"<<endl;
     }
};
```

```
int main(void)
{
   Employee e1; //creating an object of Employee
   Employee e2;
   return 0;
}
```

1.15 file stream in c++

The fstream library allows us to work with files.

To use the fstream library, include both the standard <iostream> AND the <fstream> header file:

Class	Description
Ofstream	create and writes to file
Ifstream	reads form files
Fstream	A combination of ofstream and ifstream: creates, reads, and writes to files

example-

```
 #include <fstream>
#include <iostream>
using namespace std;
int main () {
  char input[75];
  ofstream os;
  os.open("testout.txt");
  cout <<"Writing to a text file:" << endl;
  cout << "Please Enter your name: ";
  cin.getline(input, 100);
  os << input << endl;
  cout << "Please Enter your age: ";
  cin >> input;
  cin.ignore();
```

```
  os << input << endl;
  os.close();
  ifstream is;
  string line;
  is.open("testout.txt");
  cout << "Reading from a text file:" << endl;
  while (getline (is,line))     {
 cout << line << endl;
  }
  is.close();
  return 0;
}
```

Software used:

Compilers

Code::Blocks supports multiple compilers, including GCC, MinGW, Digital Mars, Microsoft Visual C++, Borland C++, LLVM Clang, Watcom, LCC and the Intel C++ compiler. Although the IDE was designed for the C++ language, there is some support for other languages, including Fortran and D. A plug-in system is included to support other programming languages.

Code editor

The IDE features syntax highlighting and code folding (through its Scintilla editor component), C++ code completion, class browser, a hex editor and many other utilities. Opened files are organized into tabs. The code editor supports font and font size selection and personalized syntax highlighting colours.

Debugger

The Code::Blocks debugger has full breakpoint support. It also allows the user to debug their program by having access to the local function symbol and argument display, user-defined watches, call stack, disassembly, custom memory dump, thread switching, CPU registers and GNU Debugger Interface.

User migration

Some of Code::Blocks features are targeted at users migrating from other IDE's - these include Dev-C++, Microsoft Visual C++ project import (MSVC 7 & 10), and Dev-C++ Devpak support.

Project files and build system

Code::Blocks uses a custom build system, which stores its information in XML-based project files. It can optionally use external makefiles, which simplifies interfacing with projects using the GNU or qmake build systems.

Conclusion