Deemah Almofarreh, Tejashri Parurkar, Ryan Buraus, Yao Shen

Professor Llesh Miraj

Principles of Programming Languages

12/9/19

C++ Programming Language

**Project Report**

For our project on the C++ programming language, we researched the following aspects of the language: the paradigms, history, evolution, TIOBE index, Google Trends, use of the language in the industry, programming problems it’s intended for, and main and specific features of the language. We also created and explained some code in C++, further displaying the output.

**Project Description and User Manual**

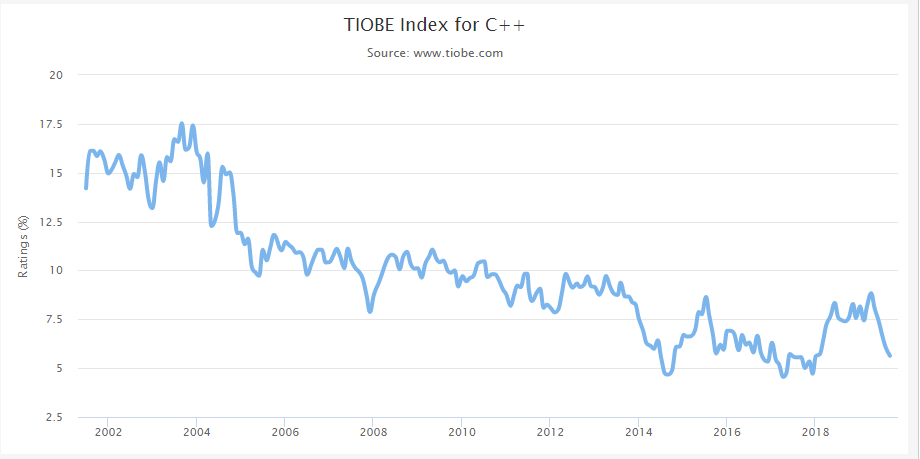
Paradigms:

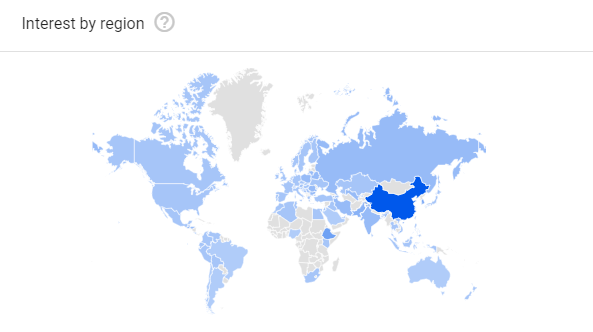
* A programming paradigm is a model of programming based on distinct concepts that shapes the way programmers design, organize and write programs. C++ is a multi-paradigm language. C++ is an example of a procedural programming language that’s statically typed. It’s object oriented and uses parallel processing.

History and Evolution:

* The Stimula 67 language[1979]
* C with classes
* Cfront - a self hosting compiler
* C++[1983]
* Commercial product, Book.[1985]
* Borland’s Turbo C++ launch[‘90]
* C++98/ISO std published[1998]
* and C++03 Respond queries[‘03]
* Released Technical Report[‘05]
* New C++ std[mid-2011]
* C++14[2014]
* C++17[2017]
* C++20… Being developed

TIOBE Index and Google Trends:





Use of C++ in the Industry:

* Compilers: used to develop languages such as Java and C# are mainly written in C++.
* Databases: used to develop database software such as MySQL and Postgres.
* Graphics: used in building real-time, image processing, mobile sensor applications, as well as visual effects.
* Libraries: ML libraries, such as Tensorflow, use C++ in the backend because of its speed.
* Web browsers: used to develop rendering engines of various web browsers.

Programming Problems It’s Intended For: C++ is intended for programming problems that require:

* scalability
* exception handling
* compatibility with C
* function overloading
* classes and inheritance
* control over the memory
* reusable and maintainable code

Main Features of C++:

* Object-oriented: it follows the concepts of OOP such as polymorphism, inheritance, encapsulation, abstraction.
* Rich library support: through C++ (STL) many functions are available like sets, maps, hash tables, etc. that help in quickly writing code.
* DMA (Dynamic Memory Allocation): since C++ supports the use of pointers, it allows us to allocate memory dynamically.
* Compiled: C++ compiles directly to a machine’s native code.
* Portable: C++ has a wide range of compilers that run on many different platforms that support it. Code that exclusively uses C++'s standard library will run on many platforms with few to no changes.
* Compiler-based and speed: C++ is compiler-based, hence, it is much faster than other programming languages.

Specific Features of C++:

* Auto:

#include <bits/stdc++.h>

using namespace std;

int main()

{

auto x = 4;

auto y = 3.37;

auto ptr = &x;

cout << typeid(x).name() << endl

<< typeid(y).name() << endl

<< typeid(ptr).name() << endl;

return 0;

}

Code:

* Insertion Sort:

#include<iostream>

using namespace std;

int main()

{

int i,j,n,temp,a[30];

cout<<"Enter the number of elements:";

cin>>n;

cout<<"\nEnter the elements\n";

for(i=0;i<n;i++)

{

cin>>a[i];

}

for(i=1;i<=n-1;i++)

{

temp=a[i];

j=i-1;

while((temp<a[j])&&(j>=0))

{

a[j+1]=a[j]; //moves element forward

j=j-1;

}

a[j+1]=temp; //insert element in proper place

}

cout<<"\nSorted list is as follows\n";

for(i=0;i<n;i++)

{

cout<<a[i]<<" ";

}

return 0;

}

**Project Process Description**

Team role assignments:

* Tejashri explained the paradigms, the history, and the evolution of C++. She also showed and explained the code that demonstrated insertion sort in C++, further displaying the output.
* Ryan explained the use of C++ in the industry and the programming problems it’s intended for.
* Yao showed and explained the TIOBE index and Google Trends for C++. He also explained the features specific to C++.
* Deemah explained the main features of C++.