

प्रोग्रामिंग भाषा

(Programming Language)

प्रोग्रामिंग भाषा, कंप्यूटर को बताने के लिए इस्तेमाल होने वाली एक "भाषा" है कि क्या करना है। यह निर्देशों का एक सेट है जिसे कंप्यूटर समझ सकता है और निष्पादित कर सकता है।

विस्तार में:

प्रोग्रामिंग भाषाएं, मनुष्यों को कंप्यूटर के साथ संवाद करने की अनुमति देती हैं। कंप्यूटर हमारी भाषा नहीं समझते, इसलिए हमें उन्हें "समझाने" के लिए एक ऐसी भाषा का उपयोग करना होता है जो उन्हें समझ में आए।

यहां कुछ मुख्य बातें हैं जो प्रोग्रामिंग भाषाओं को खास बनाती हैं:

- **निर्देशों का सेट:**

प्रोग्रामिंग भाषाएं, कंप्यूटर को विशिष्ट कार्य करने के लिए निर्देश देने के लिए उपयोग किए जाने वाले शब्दों, प्रतीकों और नियमों का एक सेट होती हैं।

- **वाक्यविन्यास (Syntax):**

प्रत्येक प्रोग्रामिंग भाषा का अपना व्याकरण होता है, जिसे वाक्यविन्यास कहा जाता है। यह निर्धारित करता है कि निर्देशों को कैसे लिखना है।

- **अर्थ (Semantics):**

अर्थ, निर्देशों के वास्तविक अर्थ को संदर्भित करता है। यह बताता है कि कंप्यूटर क्या करेगा जब उसे एक विशिष्ट निर्देश प्राप्त होगा।

- **प्रकार प्रणाली (Type System):**

कई प्रोग्रामिंग भाषाएं, चर (variables) और डेटा के प्रकारों को परिभाषित करती हैं, जो त्रुटियों को रोकने और कोड को अधिक विश्वसनीय बनाने में मदद करती हैं।

उदाहरण के लिए:

- वेबसाइट बनाने के लिए, आपको HTML, CSS और JavaScript जैसी भाषाओं का उपयोग करना होगा।
 - मोबाइल ऐप्स बनाने के लिए, आपको Java, Kotlin (Android के लिए) या Swift (iOS के लिए) जैसी भाषाओं का उपयोग करना होगा।
 - डेटा विश्लेषण के लिए, आपको Python, R या SQL जैसी भाषाओं का उपयोग करना होगा।
- प्रोग्रामिंग भाषाएं, सॉफ्टवेयर, वेबसाइट, ऐप्स और बहुत कुछ बनाने का एक अनिवार्य हिस्सा हैं।

C++ Language

C++ एक शक्तिशाली और लोकप्रिय प्रोग्रामिंग भाषा है, जो ऑब्जेक्ट-ओरिएंटेड प्रोग्रामिंग (Object-Oriented Programming) का समर्थन करती है। यह सी भाषा का एक विस्तार है और सिस्टम प्रोग्रामिंग, गेम डेवलपमेंट, और विभिन्न प्रकार के अनुप्रयोगों में व्यापक रूप से उपयोग की जाती है।

C++ क्या है?

C++ एक मध्य-स्तरीय प्रोग्रामिंग भाषा है, जिसका अर्थ है कि यह उच्च-स्तरीय (जैसे, अंग्रेजी के करीब) और निम्न-स्तरीय (जैसे, मशीन कोड के करीब) दोनों प्रकार की प्रोग्रामिंग सुविधाओं का संयोजन प्रदान करती है। इसे 1979 में Bjarne Stroustrup द्वारा Bell Labs में विकसित किया गया था, और इसका नाम पहले "C with Classes" था, जिसे बाद में 1983 में C++ कर दिया गया, [Wikipedia](#). यह एक सामान्य-उद्देश्यीय भाषा है, जिसका अर्थ है कि इसका उपयोग विभिन्न प्रकार के कार्यों के लिए किया जा सकता है।

C++ का उपयोग कहाँ होता है?

- **सिस्टम प्रोग्रामिंग:**

C++ का उपयोग ऑपरेटिंग सिस्टम, ड्राइवर, और अन्य सिस्टम-लेवल सॉफ्टवेयर बनाने के लिए किया जाता है, [TechTarget](#).

- **गेम डेवलपमेंट:**

C++ अपनी गति और प्रदर्शन के कारण गेम डेवलपमेंट में एक लोकप्रिय विकल्प है, YouTube.

- **एप्लिकेशन डेवलपमेंट:**

C++ का उपयोग डेस्कटॉप एप्लिकेशन, वेब ब्राउज़र, और अन्य प्रकार के अनुप्रयोग बनाने के लिए किया जाता है, [Study.com](#).

- **डेटाबेस:**

C++ का उपयोग डेटाबेस सिस्टम बनाने और उन पर काम करने के लिए किया जाता है

#include

C++ में, `#include` एक प्रीप्रोसेसर निर्देश है जो किसी प्रोग्राम में हेडर फाइलों को शामिल करने के लिए उपयोग किया जाता है। इसका मतलब है कि आप किसी अन्य फाइल (हेडर फाइल) की सामग्री को अपने वर्तमान C++ कोड में कॉपी और पेस्ट कर रहे हैं। इससे आपको उस हेडर फाइल में परिभाषित फंक्शन, क्लासेस, और अन्य तत्वों का उपयोग करने की अनुमति मिलती है।

उदाहरण के लिए, `#include <iostream>` का उपयोग करके, आप इनपुट/आउटपुट स्ट्रीम (जैसे `cin` और `cout`) तक पहुंचने के लिए आवश्यक जानकारी शामिल करते हैं।

यहां कुछ मुख्य बातें हैं:

- **हेडर फाइलें:**

हेडर फाइलें (जैसे `<iostream>`, `<cmath>`, `<string>`) में अक्सर फंक्शन, क्लास, और अन्य तत्वों की परिभाषाएँ होती हैं जो सामान्य कार्यों के लिए उपयोग की जाती हैं।

- **प्रीप्रोसेसर निर्देश:**

`#include` एक प्रीप्रोसेसर निर्देश है, जिसका मतलब है कि यह कंपाइलेशन (संकलन) प्रक्रिया के दौरान निष्पादित होता है, न कि प्रोग्राम के चलने के दौरान।

- **उपयोग:**

`#include` का उपयोग करके, आप अपने कोड को अधिक व्यवस्थित और पठनीय बना सकते हैं, क्योंकि आप सामान्य कार्यों के लिए आवश्यक कोड को एक अलग फाइल में रख सकते हैं और फिर उसे अपने मुख्य प्रोग्राम में शामिल कर सकते हैं।

- **उदाहरण:**

- `#include <iostream>`: इनपुट/आउटपुट के लिए।
- `#include <cmath>`: गणितीय कार्यों के लिए।
- `#include <string>`: स्ट्रिंग हेरफेर के लिए।
- `#include <conio>`: स्ट्रिंग हेरफेर के लिए।

Iostream.h

C++ में, `iostream` एक हेडर फ़ाइल है जो इनपुट और आउटपुट (I/O) संचालन के लिए आवश्यक कार्यों को परिभाषित करती है। यह लाइब्रेरी C++ प्रोग्राम में उपयोगकर्ता से डेटा प्राप्त करने (जैसे कीबोर्ड से) और स्क्रीन पर जानकारी प्रदर्शित करने (जैसे कंसोल पर) के लिए आवश्यक है। `iostream` का मतलब है "इनपुट/आउटपुट स्ट्रीम"।

विस्तार से:

- हेडर फ़ाइल:

C++ में, `iostream` एक हेडर फ़ाइल है जिसे प्रोग्राम में शामिल करने की आवश्यकता होती है ताकि आप `cin` (इनपुट के लिए) और `cout` (आउटपुट के लिए) जैसे इनपुट/आउटपुट स्ट्रीम ऑब्जेक्ट का उपयोग कर सकें।

- इनपुट/आउटपुट स्ट्रीम:

`iostream` लाइब्रेरी ऑब्जेक्ट-ओरिएंटेड तरीके से इनपुट और आउटपुट प्रदान करती है। यह बाइट्स के क्रम (स्ट्रीम) के रूप में डेटा को संभालती है, जिसे आप किसी डिवाइस (जैसे कि कीबोर्ड या स्क्रीन) से पढ़ने या उस पर लिखने के लिए उपयोग कर सकते हैं।

- मुख्य उपयोग:

`iostream` का उपयोग मुख्य रूप से दो कार्यों के लिए किया जाता है:

- इनपुट: कीबोर्ड जैसे इनपुट डिवाइस से डेटा प्राप्त करने के लिए `cin` का उपयोग करें।
- आउटपुट: स्क्रीन जैसे आउटपुट डिवाइस पर जानकारी प्रदर्शित करने के लिए `cout` का उपयोग करें।

Conio.h

C++ में `conio.h` एक हेडर फ़ाइल है जो कंसोल इनपुट और आउटपुट (console input/output) के लिए फ़ंक्शन प्रदान करती है, जैसे कि `getch()`, `clrscr()` और `gotoxy()`। यह C और C++ दोनों में उपयोग होती है, लेकिन यह C++ का मानक हेडर नहीं है। यह मुख्य रूप से DOS और Windows जैसे ऑपरेटिंग सिस्टम में उपयोग किया जाता है।

`conio.h` का पूरा नाम "console input/output" है। यह हेडर फ़ाइल कंसोल-आधारित अनुप्रयोगों के लिए उपयोगी है, जो टेक्स्ट-आधारित इंटरैक्शन प्रदान करते हैं। उदाहरण के लिए, `getch()` फ़ंक्शन का उपयोग एक अक्षर इनपुट लेने के लिए किया जाता है, जबकि `clrscr()` फ़ंक्शन का उपयोग कंसोल स्क्रीन को साफ़ करने के लिए किया जाता है।

हालांकि, यह ध्यान रखना महत्वपूर्ण है कि `conio.h` C++ का एक मानक हेडर नहीं है, और यह सभी कंपाइलरों द्वारा समर्थित नहीं है। विशेष रूप से, GCC और Clang जैसे आधुनिक कंपाइलर `conio.h` का समर्थन नहीं करते हैं। यदि आप पोर्टेबल C++ कोड लिखना चाहते हैं, तो आपको `conio.h` के बजाय अन्य हेडर फ़ाइलों और फ़ंक्शन का उपयोग करना चाहिए, जैसे कि `<iostream>` और `<conio.h>` के बजाय `<windows.h>` (Windows के लिए)।

उदाहरण के लिए, यदि आप Dev C++ जैसे कंपाइलर का उपयोग कर रहे हैं, तो आप `clrscr()` फ़ंक्शन के लिए `system("cls")` का उपयोग कर सकते हैं, जो विंडोज कंसोल को साफ़ करने के लिए एक सिस्टम कॉल है, या `<windows.h>` हेडर फ़ाइल शामिल कर सकते हैं और `system("cls")` का उपयोग कर सकते हैं। इसके अलावा, यदि आप `getch()` फ़ंक्शन का उपयोग कर रहे हैं, तो आप `<conio.h>` के बजाय `<iostream>` और `std::cin.get()` का उपयोग कर सकते हैं।

Void Main

C++ में, `void main()` एक फ़ंक्शन है जिसे प्रोग्राम के निष्पादन का प्रारंभिक बिंदु माना जाता है। `void` का मतलब है कि यह फ़ंक्शन कुछ भी वापस नहीं करता है, और `main` फ़ंक्शन का नाम है। C++ में, `main` फ़ंक्शन प्रोग्राम का प्रवेश बिंदु होता है।

विस्तार में:

- **void:**

यह कीवर्ड दर्शाता है कि फ़ंक्शन कोई मान वापस नहीं करेगा। इसका मतलब है कि जब `main` फ़ंक्शन निष्पादित हो जाता है, तो यह कोई मान वापस नहीं भेजेगा।

- **main:**

यह C++ प्रोग्राम में मुख्य फ़ंक्शन का नाम है। यह वह जगह है जहां प्रोग्राम का निष्पादन शुरू होता है।

- **प्रोग्राम का प्रवेश बिंदु:**

जब आप C++ प्रोग्राम चलाते हैं, तो ऑपरेटिंग सिस्टम `main` फ़ंक्शन को ढूंढता है और उसे निष्पादित करता है।

Variable

C++ में, एक वेरिएबल (variable) एक नाम दिया गया मेमोरी स्थान (memory location) है जिसका उपयोग प्रोग्राम में डेटा संग्रहीत करने के लिए किया जाता है। यह एक कंटेनर की तरह है जो प्रोग्राम के दौरान बदल सकने वाली (variable) वैल्यू को स्टोर करता है।

विस्तार में:

- **वेरिएबल क्या है?**

वेरिएबल एक नाम है जो कंप्यूटर की मेमोरी में एक विशिष्ट स्थान को संदर्भित करता है।

- **डेटा स्टोरेज:**

इस स्थान का उपयोग डेटा (जैसे संख्या, टेक्स्ट, या अन्य मान) को संग्रहीत करने के लिए किया जाता है।

- **मान बदलना:**

प्रोग्राम के चलने के दौरान, वेरिएबल में संग्रहीत मान को बदला जा सकता है।

- **डेटा प्रकार:**

प्रत्येक वेरिएबल का एक डेटा प्रकार होता है, जो यह निर्धारित करता है कि किस प्रकार का डेटा संग्रहीत किया जा सकता है (जैसे पूर्णांक, फ्लोट, कैरेक्टर, आदि)।

- **उदाहरण:**

मान लीजिए कि आप एक प्रोग्राम में एक उपयोगकर्ता की उम्र संग्रहीत करना चाहते हैं। आप एक वेरिएबल (जैसे age) बना सकते हैं और उसमें उपयोगकर्ता की उम्र (जैसे 25) संग्रहीत कर सकते हैं। फिर, आप प्रोग्राम में age वेरिएबल का उपयोग करके उस मान को एक्सेस या संशोधित कर सकते हैं।

- **वेरिएबल नाम:**

वेरिएबल को एक विशिष्ट नाम दिया जाता है ताकि प्रोग्रामर उस स्थान को पहचान सके। उदाहरण के लिए, age, name, price जैसे नाम वेरिएबल के लिए उपयोग किए जा सकते हैं।

Data Type

C++ में डेटा टाइप (Data Type) एक विशेषता है जो बताती है कि किसी वेरिएबल (variable) में किस तरह का डेटा संग्रहीत किया जा सकता है। यह कंपाइलर को यह समझने में मदद करता है कि डेटा को कैसे इंटरप्रेट करना है और उस पर कौन से ऑपरेशंस किए जा सकते हैं। दूसरे शब्दों में, डेटा टाइप यह निर्धारित करता है कि वेरिएबल मेमोरी में कितनी जगह लेगा और उस मेमोरी में किस तरह का डेटा स्टोर होगा।

C++ में डेटा टाइप को मोटे तौर पर दो श्रेणियों में बांटा जा सकता है:

1. 1. प्रिमिटिव (Primitive) या बिल्ट-इन (Built-in) डेटा टाइप:

ये C++ में पहले से मौजूद डेटा टाइप हैं, जैसे:

- **int:** पूर्णांक संख्याओं (whole numbers) के लिए, जैसे 10, -5, 0।
- **float:** दशमलव संख्याओं (decimal numbers) के लिए, जैसे 3.14, -2.5।
- **double:** डबल-प्रेसिजन (double-precision) फ्लोटिंग-पॉइंट संख्याओं के लिए, जो फ्लोट से अधिक सटीक होती हैं।
- **char:** एक अक्षर या वर्ण (character) के लिए, जैसे 'A', 'z', '5'।
- **bool:** सत्य (true) या असत्य (false) मानों के लिए।
- **void:** कुछ मामलों में, जैसे कि फ़ंक्शन जो कोई मान वापस नहीं करता है।

2. 2. व्युत्पन्न (Derived) डेटा टाइप:

ये प्रिमिटिव डेटा टाइप से बनाए जाते हैं, जैसे:

- **एरे (Arrays):** एक ही प्रकार के डेटा के कई मानों को संग्रहीत करने के लिए।
- **पॉइंटर्स (Pointers):** मेमोरी एड्रेस को संग्रहीत करने के लिए।
- **रेफरेंस (References):** किसी अन्य वेरिएबल को संदर्भित करने के लिए।
- **क्लासेस (Classes) और स्ट्रक्चर्स (Structures):** डेटा और फ़ंक्शन का एक समूह बनाने के लिए।
डेटा टाइप का उपयोग चर (variables) और स्थिरांक (constants) को परिभाषित करने के लिए किया जाता है ताकि कंपाइलर को यह पता चल सके कि डेटा को कैसे संग्रहीत और संसाधित करना है।