مقدمه

تاریخچه
ساختار ساده یک برنامه
برخی از ویژگی های زبان
مراحل پیاده سازی برنامه

تاریخچه

C++ بر مبنای زبان برنامه نویسی C است. زبان C در سال ۱۹۷۲ در آزمایشگاه Bell Telephone نوسط Dennis Ritchie به عنوان زبان پیاده سازی برای سیستم عامل یونیکس طراحی شد. مقدار زیادی از برنامه نویسی یونیکس با زبان C انجام شده است C .در نتیجه تکوین پروسه ای است که با یک زبان قدیمی تر به نام BCPL شروع شده بود. زبان BCPL زبانی بر اساس زبان B بوده است که توسط Ken Thompson در آزمایشگاه Bell طراحی شده بود.

به دلیل اینکه C زبان قدرتمند و انعطاف پذیری بود، سریعا گسترش پیدا کرد. برنامه نویسان شروع به استفاده از آن برای انواع برنامه ها کردند .سازمان های مختلف شروع به پیاده سازی نسخه های C خود شدند. تا اینکه در سال ۱۹۸۳، ANSIاستاندارد C را تنظیم کرد که به عنوان ANSI Standard C شناخته می شود. کامپایلرهای بعدی C از این استاندارد پیروی کردند .

زبان برنامه نویسي C++ بر اساس زبان C توسط Bjarne Stroustrup ابداع شد. آنچه امروزه C++ نامیده مي شود از سال ۱۹۷۹ آغاز شده است. نسخه اولیه آن "C with classes" نامیده شد که بعدا به C++ تغییر کرد C++. کلیه ویژگی های زبان C را داراست. تفاوت بین آنها اینستکه C++ شي گرائی را پشتیبانی می کند. البته پیشرفت های دیگری هم دارد برای مثال کار با رشته ها و سروکارداشتن با خطاها در آن قوی تر است.

نسخه اول C++ ابتدا در AT&T در سال ۱۹۸۳ استفاده شد. اولین نسخه تجاری آن در اکتبر ۱۹۸۵ به بازار آمد. در سال ۱۹۹۸ ANSIو ISO متفقا C++ را استاندارد کردند. به همین دلیل اغلب C++ محض را ANSI Standard C++ یا ISO Standard C++ می نامند .

با تکامل C++ یک کتابخانه استاندارد هم با آن شکل گرفت. اولین کتابخانه استاندارد C++ کتابخانه stream I/O بود که امکاناتی برای جایگزینی توابع قدیمی C مانند printf و scanf مهیا کرد. بعد از آن مهمترین کتابخانه استاندارد کتابخانه Standard Template بود.

توجه داشته باشید که کد C در کامپایلر C++ کامپایل می شود اما عکس آن صادق نیست و کد C++ لزوما در کامپایلر C کامپایل نمی شود .

ساختار ساده یک برنامه

برای آشنائی سریعتر با ساختار برنامه C++ یک برنامه ساده کوچک در اینجا ارائه می شود که ممکن است در وحله اول کاملا قابل درک نباشد. توضیحات بیشتر درباره برنامه در دنباله آن داده شده است.

مثال. برنامه Hello.cpp یک پیغام را روی صفحه نمایش می دهد .

```
/* HELLO.CPP */
#include <iostream>
using namespace std;
int main()
{
  //print hello word on the screen
  cout << "Hello World!\n";
  return 0;
}
```

تابع اصلی

در زبان C برنامه از یک سری تابع (function) تشکیل می شود. هر برنامه الزاما یک تابع اصلی به نام main دارد که برنامه از آن شروع به اجرا می کند .

نوع برگشتی تابع در خط اعلان تابع و قبل از نام تابع تعیین می شود. که برای تابع اصلی معمولا نوع صحیح یا int درنظر گرفته می شود. چون برخی از سیستم عامل ها نیازدارند برنامه مقداری را به آنها برگرداند تابع main یک عدد صحیح را بر می گرداند تا سیستم عامل متوجه بشود که همه چیز درست است .البته در بعضی موارد تابع اصلی را می توان به صورت void main() هم تعریف کرد. کلمه void مشخص می کند تابع چیزی بر نمی گرداند.

مقداری که تابع بر می گرداند در مقابل دستور return نوشته می شود. اگر تابعی مقداری برنگرداند و از نوع void تعریف شده باشد نیازی به استفاده از دستور return در تابع نیست .

بدنه توابع و بلوک های کد مابین دو کروشه باز و بسته { } قرار می گیرند. کروشه مرز یک بلاک از کد را معین می کند. وقت کنید که برای هر کروشه باز { یک کروشه بسته } باید وجود داشته باشد .

```
int main()
{
  return 0;
}
```

فایل های ضمیمه

معمولا اولین عمل در هر برنامه C++ ضمیمه کردن فایل های هدر است. فایل های هدر (include file) فایل هائی شامل تعاریف توابع یا کلاس ها هستند .اگر توابع یا متغیرهائی وجود دارند که در برنامه های مختلف مورد استفاده قرار می گیرند بهتر است در یک فایل هدر قرار داده شوند، سپس فایل هدر در هرجائی که از آن استفاده می شود ضمیمه شود .

برای ضمیمه کردن یک فایل به برنامه علامت # سپس include و نام فایل ضمیمه در داخل < > ذکر می شود.

اولین خط برنامه Hello.cpp فایل هدر iostream.h را ضمیمه برنامه می کند .

```
#include <iostream.h>
```

تعداد زیادی از فایل های هدر در C++ برای استفاده از توابع کتابخانه ای همراه با نصب کامپایلر در اختیار برنامه نویس قرار می گیرند. یکی از این فایل ها فایل هدر iostream.h است که توابعی برای عملیات ورودی و خروجی دارد. اگر این فایل در ابتدای برنامه ضمیمه شود کلیه کلاس ها و توابع تعریف شده در آن قابل دسترسی هستند .

پسوند h. مشخص مي كند يک فایل هدر است. اگر کامپایلر C را استفاده می کنید پسوند h. باید مشخص شود ولی در C++ نیاز نیست. البته بهتر است که ذکر شود تا توسط کامپایلرها پشتیبانی شود.

نمایش پیغام

تابع cout برای نمایش داده ها و پیغام ها روی صفحه نمایش بکار می رود .این توابع در فایل هدر iostream قرار دارد که باید به برنامه ضمیمه شده باشد .

```
cout << "Hello World!\n";
```

در زبان C++ یک رشته کاراکتری باید بین دو علامت " " محصور شود

\n به کامپایلر می گوید از یک خط جدید شروع کند.

فضای اسمی

فضای اسمی (namespace) اجازه گروه بندی مجموعه ای از توابع یا اشیای سراسری را تحت یک نام می دهد. وقتی یک فایل هدر اضافه می شود محتویات آن در فضای اسمی استاندارد std قرار می گیرد. برای دسترسی به کتابخانه های استاندارد درون stdباید از عملگر :: استفاده شود .

مثال. وقتی iostream در برنامه ضمیمه شده است برای دسترسی به تابع cout از این کتابخانه باید فضای اسمی std قبل از آن و بدنبالش علامت :: تعیین شود:

```
Std::cout <<"Hello World!\n";
```

البته به سادگی می توان با استفاده از عبارت using فضای اسمی مورد استفاده را در برنامه تعیین کرد و به عناصر درون std تنها با صدازدن نامشان دسترسی پیدا کرد.

وقتی پسوند h. اضافه می شود نیازی به اضافه کردن فضای اسمی استاندارد نیست ولی به طور معمول بهتر است که از استاندارد ANSI/ISOتبعیت بشود و فضای اسمی std توسط عبارت using تعیین شود .

نکته. کامپایلرهای قدیمی عبارت using را قبول نمی کنند.

توضیحات

توضیحات (comment) عباراتی هستند که هنگام ترجمه نادیده گرفته می شوند .توضیحات شرح مرحله به مرحله از عملیات برنامه را برای برنامه نویس، یا هر فردی که روی کد برنامه کار می کند، ارائه می دهد.

در C++ دو روش برای ساختن توضیح وجود دارد؛ یک روش که برای توضیحات کوتاه است با علامت // شروع می شود. کامپایلر از این علامت تا انتهای خط را به عنوان توضیح درنظر می گیرد .

روش دیگر که معمولا برای توضیحات طولانی تر استفاده می شود بستن توضیحات بین علائم /* */ است.

```
//print hello word on the screen
/* This is my first program in C++ */
```

نکته. توضیحات یک خطی به دنبال علامت // تنها در کامپایلرهای C++ قابل استفاده است .

برخی از ویژگی های زبان

زبان های برنامه نویسی مختلفی در دنیا وجود دارد که برای برنامه نویسی مناسب هستند. اما به چند دلیل برنامه نویسان حرفه ای Cرا ترجیح می دهند .برخی از ویژگی های زبان C/C++ عبارتند از:

- C++ زبان برنامه نویسی قدرتمند و انعطاف پذیری است و محدودیتی برای کارهائی که می توان توسط آن انجام داد وجود ندارد. و برای پروژه هائی مانند ساخت سیستم های عامل، پردازشگرهای متن، گرافیک، صفحات گسترده و حتی کامپایلرهای زبان های دیگر بکار می رود .
- انواع مختلفی از کامپایلرهای C ، C++، Visual Cو برنامه های سودمند جانبی موجود هستند .
- C++زبان قابل حملی (portable) است یعنی برنامه ای که به زبان C++ نوشته می شود با حداقل تغییرات روی هر سیستمی اجرا شود.
- زبان C++ دارای تعداد کمی کلمات کلیدی است. ولی با آن می توان هر برنامه را نوشت.
- هر عبارت در C++ به یک سمیکولن (;) ختم می شود. سمیکولن برای کامپایلر یک دستور یا عبارت از کد را مشخص می کند .
- C++حساس به متن است یعني بین حروف بزرگ و کوچک تفاوت قائل می شود. بنابراین دقت کنید int main() با int Main()یکسان نیست. خیلی از اشتباهات مبتدیان به دلیل بی توجهی به این نکته بروز می کند .

مراحل پیاده سازی برنامه

آماده سازی منطق برنامه

قبل از نوشتن برنامه باید مراحل حل مسئله را تعیین کنید. ابتدا مشکل را شناخته، راه حل مربوط به آن را پیدا کنید. سپس اقدام به نوشتن برنامه نمائید .

تهیه کد برنامه

کد برنامه (source code) مجموعه عبارات یا دستوراتی است کامپیوتر را هدایت می کند تا عمل موردنظر شما انجام بپذیرد. از یک ویرایشگر متن برای وارد کردن کد برنامه استفاده کنید. اغلب کامپایلرهای نظیر Borland's Turbo C++ و Visual C/C++همراه با یک محیط مجتمع (IDE) می آیند که اجازه تایپ، کامپایل و لینک برنامه را در یک محیط مناسب می دهند. در غیراینصورت از ویرایشگرهای متن دیگر مانندEdit ، Notepadو Microsoft Windows می توانید استفاده کنید و برنامه خود را با فرمت ASCII روی دیسک ذخیره نمائید .

کد برنامه را با پسوند .cpp ذخیره کنید .

ترجمه کد برنامه

برای ترجمه یک برنامه C/C++ کامپایلرهای مختلفی وجود دارد. نسخه رایگان کامپایلر Borland C++ محیط مجتمع پیاده سازی ندارد و یک کامپایلر خط فرمانی است یعنی باید در محیط سیستم عامل DOS فرمان ترجمه برنامه صادر شود .

مثال. اگر Borland's Turbo C++ را استفاده می کنید، برای ترجمه برنامه Hello.cpp فرمان زیر را در خط فرمان سیستم عامل باید وارد نمائید:

```
bcc Hello.cpp
```

ترجمه برنامه در کامپایلرهای با محیط مجتمع (مانند Borland C++ 3.1) یا محیط گرافیکی (مانندMicrosoft Visual C++) راحت تر صورت می گیرد. کافی است ازمنو گزینه Compile یا Run را انتخاب کنید .

اگر کامپایلر خطائی در برنامه مشاهده نکرد فایل مقصد حاوی کد زبان ماشین برنامه هم نام برنامه و با پسوند .obj ایجاد می شود. اگر خطائی در برنامه وجود داشته باشد گزارش خطا توسط کامپایلر تهیه می شود در اینصورت باید به کد برنامه مراجعه کرده خطا را برطرف کنید .

تهیه برنامه اجرائی

وقتی از توابع کتابخانه ای استفاده می کنید، فایل مقصد تولید شده باید با کد مقصد تابع کتابخانه ترکیب شود تا فایل اجرائی نهائی شکل بگیرد. این فرآیند linking نام دارد که توسط برنامه لینکر (linker) انجام می گیرد .اگر لینکر در این فرآیند با مشکلی مواجه نشود، یک برنامه اجرائی روی دیسک هم نام برنامه و با پسوند .exe ایجاد می شود .

اجرای برنامه

با تولید برنامه اجرائی، می توانید آنرا اجرا کنید. باید برنامه را تست کنید تا مطمئن شوید درست کار می کند. اگر نتایجی متفاوت از آنچه انتظار داشتید دریافت کردید باید به قدم اول برگشته علت خطا را پیدا کنید و کد برنامه را اصلاح کنید و مجددا برنامه را ترجمه، لینک و اجرا کنید.
در اکثر کامپایلرها امکان انجام مراحل ترجمه، لینک و اجرا در یک مرحله وجود دارد، گرچه در اینجا به عنوان مراحل جداگانه مطرح شد.

مثال. اگر برنامه Hello.cpp موفق ترجمه و لینک شود فایل های Hello.obj که شامل کد زبان ماشین برنامه و Hello.exe که برنامه اجرائی تولید شده است روی دیسک ایجاد می شود. با اجرای برنامه جمله Hello, World! روی صفحه مشاهده می شود.