* **MD5**

این الگوریتم یک مقدار هش 128 بیتی تولید می‌کند. این الگوریتم برای استفاده در رمزنگاری طراحی شده بود، اما با زمان ضعف‌هایی در آن کشف شدند و دیگر برای این منظور توصیه نمی‌شود. [اما هنوز برای تقسیم‌بندی پایگاه داده و محاسبه checksum برای اعتبارسنجی انتقال فایل‌ها استفاده می‌شود](https://heimdalsecurity.com/blog/what-is-hashing/" \t "_blank).

* **SHA-1**

این الگوریتم نخستین نسخه از استاندارد الگوریتم هش امن (SHA) است. این الگوریتم در حالی که MD5 یک مقدار هش 128 بیتی تولید می‌کند، یک مقدار هش 160 بیتی (20 بایت) تولید می‌کند. در قالب هگزادسیمال، این یک عدد صحیح 40 رقمی است. مانند MD5، این الگوریتم برای کاربردهای رمزنگاری طراحی شده بود، اما به زودی ضعف‌هایی در آن نیز شناسایی شدند. [از امروز، این الگوریتم دیگر مقاومت کمتری در برابر حمله نسبت به MD5 ندارد](https://www.freecodecamp.org/news/md5-vs-sha-1-vs-sha-2-which-is-the-most-secure-encryption-hash-and-how-to-check-them/" \t "_blank).

* **SHA-2**

این الگوریتم یک خانواده از الگوریتم‌های هش امن است که پس از SHA-1 طراحی شدند. این الگوریتم‌ها مقادیر هش با طول‌های مختلفی را تولید می‌کنند، از جمله 224، 256، 384 و 512 بیت. [این الگوریتم‌ها مقاومت بیشتری در برابر حملات رمزنگاری دارند و برای امنیت اطلاعات و ارتباطات استفاده می‌شوند](https://heimdalsecurity.com/blog/what-is-hashing/" \t "_blank).

* **NTLM**

این الگوریتم یک پروتکل احراز هویت چالش-پاسخ است که توسط مایکروسافت برای سیستم‌های ویندوز طراحی شده است. این الگوریتم از الگوریتم MD4 برای تولید مقادیر هش از گذرواژه‌های کاربران استفاده می‌کند. این الگوریتم در سه نسخه مختلف وجود دارد NTLMv1، NTLMv2 و NTLM2 Session. [این الگوریتم نیز ضعف‌های امنیتی دارد و برای جایگزینی آن با پروتکل‌های احراز هویت مدرن‌تر توصیه می‌شود](https://en.wikipedia.org/wiki/NTLM" \t "_blank).

* **LANMAN**

این الگوریتم یک الگوریتم هش قدیمی است که توسط مایکروسافت و آی‌بی‌ام برای شبکه‌های محلی طراحی شده است. این الگوریتم از الگوریتم DES برای تولید مقادیر هش از گذرواژه‌های کاربران استفاده می‌کند. این الگوریتم بسیار ضعیف است و به راحتی قابل شکستن است. [این الگوریتم دیگر در ویندوز‌های جدید پشتیبانی نمی‌شود](https://en.wikipedia.org/wiki/NTLM" \t "_blank).

**نحوه کار MD5**

MD5 یک الگوریتم هش کردن (Hash function) می باشد. در الگوریتم های هش برای همه ی ورودی های با طول متفاوت، طول کد تولید شده ثابت و غیرقابل بازگشت می باشد. بنابراین در حقیقت روشن نیست که یک عبارت معادل یک رشته MD5 چیست . در MD5 نیز فایل های ورودی با هر اندازه و هر فرمتی که باشند، کد خروجی همیشه برابر با ۱۲۸ بیت (۱۶ بایت) خواهد بود که معمولا آن را در قالب ۳۲ عدد هگزادسیمال (مبنای ۱۶) نمایش می دهند.

ویژگی دیگر این الگوریتم غیر قابل بازگشت بودن یا یک طرفه بودن کد تولید شده می باشد. یعنی نمی توان با استفاده از کد خروجی، ورودی الگوریتم را تشخیص داد. این ویژگی باعث شده است در موارد بسیاری از MD5 برای ذخیره سازی داده ها مانند ذخیره رمزهای عبور استفاده شود.

**توضیح MD5.cpp**

**متد Update**

یک بافر از بایت ها و اندازه آن را به عنوان ورودی می گیرد و وضعیت داخلی شی MD5 را با داده های جدید به روز می کند.

**متد Final**

محاسبات هش را به پایان می رساند و اعضای \_\_digest و \_\_hex\_digest را به ترتیب به صورت باینری و هگزا دسیمال به مقدار هش به دست آمده تنظیم می کند.

**متدهای digest و hex\_digest**

مقدار هش حاصل را به ترتیب به صورت باینری و هگزادسیمال برمی‌گردانند.

**متد left\_rotate**

چرخش چپ را روی یک عدد صحیح 32 بیتی با تعداد معینی از بیت انجام می دهد.

**متد apply\_md5\_on\_block**:

تابع فشرده سازی MD5 را روی یک بلوک 64 بایتی از پیام اعمال می کند.

سازنده حالت داخلی شی MD5 را با مقدار هش اولیه مقداردهی می کند.

**تابع اصلی**

نحوه استفاده از کلاس MD5 را برای محاسبه هش یک رشته نشان می دهد. یک شی MD5 ایجاد می کند، آن را با رشته «علیرضا» به روز می کند، و سپس Final را فراخوانی می کند تا مقدار هش حاصل را محاسبه و چاپ کند.