SRS

SOFTWARE REQUIREMENT SPECIFICATION

Validation of Credit Card Numbers Using the C++ Programming Language

Validation of Credit Card Numbers Using the C++ Programming Language

Surya Pratap Singh

Sejal Jain

Priya Kishore

Abstract

The validation of credit card numbers allows us to check the authenticity of the credit card number in question or whether that credit card is valid or not for electronic transactions purposes. The most popular algorithm used for credit card number validation is called the Luhn algorithm or MOD (Modulus) 10 algorithms named after its inventor, IBM scientist Hans Peter Luhn. The Luhn algorithm popularly known as MOD 10 algorithm is a simple checksum formula designed to protect against accidental errors that normally occur number generation systems and for malicious attacks. Most credit card companies and governmental identification numbers use the algorithm as a simple method of distinguishing valid numbers from collections of random digits. It is not also intended to be a cryptographically secure hash function. It is also used to validate a variety of identification numbers such as credit card numbers, IMEI numbers, National Provider Identifier numbers in US and Canadian Social Insurance numbers. This takes a look at the Luhn algorithm or MOD 10 algorithm and its implementation in a suitable programming language called the C++ programming language. The C++ credit card validation programming also offers a graphical interface to give the user 'A user-friendly' experience. It is intended to be a simple,modern,general-purpose , object-oriented, programing language.

Keywords: validation; implementation; programming; language; algorithm; credit card

INTRODUCTION

The Luhn algorithm or MOD 10 algorithm was invented by the IBM(International Business Machines, 1960) scientist Hans Peter Luhn (July 1, 1896 – August 19, 1964) and described by the US patent No. 2,950,048 filed on January 6, 1954 and granted on August 23, 1960. The algorithm is in the public domain and is in wide use today. It is not intended to be a cryptographically secure hash function. It was designed to protect against accidental errors, not malicious attacks. Most credit card and many governmental identification numbers use the algorithm as a simple method of distinguishing valid numbers from collections of random digits. Credit card refers to an electronic miniature card that contains secret

information about the identity of the card holder. A credit card is then used by the identifier to transact business across various electronic platforms like Amazon.com and ebay.com for the purpose of acquiring goods and services. Credit cards contain vital information in the form secret codes that requires processing before payment can be made or translation before an electronic platform can authenticate and validate the credit card number before transaction takes place.

Normally for a typical credit card transaction on an electronic commerce site, several stages are involved, this include ,the customer , the merchant, the payment gateway, the acquiring bank's processor, the credit card interchange, the customer's credit card issuer , and finally the merchant acquiring bank. First of all, the first destination is the payment gateway then to the appropriate processor. The processor submits your credit card information to the credit card interchange system. After, routing your credit card information through the credit card interchange system, then you are routed again to the issuing bank where your information is verified for available funds in your credit card account after which your account is credited and then the transaction is completed. The Luhn algorithm is the most fundamental structural algorithm for validating credit card information which is used by most credit card companies even though other complex algorithms also exist for validating other types of cards.

Typically, all credit card numbers are numeric and the length of the credit card number is between 12 digits and 19 digits. Credit cards mostly issued Dinners Club contains 14, 15,or 16 digits while 15 digits relates American Express, 13 or 16 digits are handled by Visa and finally 16 digits credit card is handled by Master card.

METHODOLOGY

In this section of the paper, the methodology is divided into two phases. The purpose of phase one is to explore the 'anatomy' of credit cards and their algorithms (Luhn or MO 10) and how it is used to validate credit card numbers. In the second phase, a different kind of algorithm is designed and implemented in C++ programming language.

PHASE ONE

The anatomy of credit card

1. The first digit of the credit card is the **Major Industry Identifier (MII)**.It designates the category of the entry which issued the credit card. For example:

- 1 and 2 for Airlines
- 3 for Travel
- 4 and 5 –Banking and Financial
- 6 for Merchandising and Banking/Financial
- 7 for Petroleum
- 8 for Healthcare and Telecommunications
- 9 for National Assignment

2. **Issuer Identification Number**
The first six (6) digits are the Issuer Identification Number. It will identify the institution that issued the credit card. For example:

3. Annex – 34xxxx, 37xxxx

4. Visa – 4xxxxxx

5. Master card – 51xxxx – 55xxxx

6. Discover – 6011xx, 644xxx, 65xxxx

7. **Account Number**

When you remove the 6 identifier digits and the last digits, the remaining digits are the person's account number (7th and following excluding last digits)

8. **Check digits**

Last digit is known as the check digit or checksum. It is used to validate the credit card number using the Luhn algorithm (MOD 10 algorithm).

Background of Luhn Algorithm

The Luhn algorithm or Luhn formula, also known as the modulus 10 or MOD 10 algorithm is a simple checksum formula used to validate a variety of identification numbers, such as credit card numbers, National Provider Identifier numbers in USA and Canadian Social Insurance numbers and the rest of the world. It was created by IBM scientist Hans Peter Luhn.

When implementing electronic commerce (eCommerce) applications, it is best practice validating credit card numbers before sending it the bank for any electronic transactions.

Using the Luhn algorithm to validate Credit Card numbers

For example consider:**4012888888881881**

Step 1: Starting with the check digit double the value of every other digit (right to left every second (2^{nd}) digit.

4	0	1	2	8	8	8	8	8	8	8	8	1	8	8	1
X2		X2		X2		X2		X2		X2		X2		X2	
8		2		16		16		16		16		16		16	

Step 2: If doubling of a number results in a two digits numbers add up the digits to get a single number.

4	0	1	2	8	8	8	8	8	8	8	8	1	8	8	1
X2		X2		X2		X2		X2		X2		X2		X2	
8		2		7		7		7		7		2		7	

Step 3: Now add the undoubled digits to the odd places.

4	0	1	2	8	8	8	8	8	8	8	8	1	8	8	1
X2		X2		X2		X2		X2		X2		X2		X2	
8		2		7		7		7		7		2		7	

0	2	8	8	8	8	8	1

Step 4: Add up all the digits in this number.

4	0	1	2	8	8	8	8	8	8	8	8	1	8	8	1
X2		X2		X2		X2		X2		X2		X2		X2	
8		2		7		7		7		7		2		7	
	0		2		8		8		8		8		8		1

8 + 0 + 2 + 2 +7 + 8 + 7 + 8 + 7 + 8 + 7 + 8 + 2 + 8 + 7 + 1 = 90

If the final sum is divisible by 10, then the credit card number is valid. If it is not divisible by 10, the number is not valid.

PHASE TWO

C++ IMPLEMENTATION OF LUHN ALGORITHM (MOD 10 ALGORITHM)

Following code sample validates your credit card number against Mod 10.

```
public static bool Mod10Check(string creditCardNumber) {
//// check whether input string is null or empty
   if (string.IsNullOrEmpty(creditCardNumber)) { return false;    }

   //// 1. Starting with the check digit double the value of every other digit
   //// 2. If doubling of a number results in a two digits number, add up
   ///   the digits to get a single digit number. This will results in eight single digit numbers
   //// 3. Get the sum of the digits
   int sumOfDigits = creditCardNumber.Where((e) => e >= '0' && e <= '9')
                .Reverse()
                .Select((e, i) => ((int)e - 48) * (i % 2 == 0 ? 1 : 2))
.Sum((e) => e / 10 + e % 10);

   //// If the final sum is divisible by 10, then the credit card number
//   is valid. If it is not divisible by 10, the number is invalid.
return sumOfDigits % 10 == 0;          }
```

RESULTS

With the implementation of Luhn algorithm (MOD 10) in the C++ programming language, the program allows for any user to easily validate any kind of credit card number to ensure its validity and authenticity before making orders or purchases online. The C++ programming language is a standard programming language that cut across different platforms and so it is standard-based and allows for flexible and convenient software development methodologies .The credit card validation program is able to ascertain the validity of credit card number based on the Luhn algorithm .The process of validating the credit card numbers follows. Starting from the first digit, you double the value of every other digit (right to left, every second digit) then if by doubling of a number result in a two digit number, you add up the digits to get a single digit number. The next is for you to add the undoubled digits to the odd place/position. Furthermore, add up all the digits in this numbers and finally if the final sum is divisible by 10, then the credit card number is valid and if it is not divisible by 10, that particular credit card number is invalid.

CONCLUSION

The algorithm proposed and designed in this paper uses an open standard design or procedure and could also be implemented in another programming language lie C++ or Java. Any programming novice can easily work around the code to generate the same program using a different programming language. In the other hand, the C++ sharp credit card validation program uses a universal standard and its program that is meant to check the credibility and validity of the credit card in question and also does not detect whether or not, the true identity of the credit card holder because that is not the objective of this paper. The main aim of this paper is to design an algorithm in C++ programming language that would validate the correctness of any credit card number.

RECOMMENDATIONS

The following are the recommendations originating from this paper:

1. The Luhn or MOD 10 algorithm provides the most fundamental structure for the purpose of checking the correctness of credit card numbers.
2. The Luhn algorithm provides a suitable foundation for implementing it in an object-oriented programming language.
3. The Luhn algorithm could equally be implemented in a high-level programming language apart from the C++ programming language.
4. Any further research on this paper could be directed to detecting the true identity credit card owners using highly secured authenticating modes.

REFERENCE

Ausubel, Lawrence M., (1991), ‚The Failure of Competition in the Credit Card Market,`American Economic Review, 81(1), 50-81.

Bain, Joe S. (1956) ‚Barriers to New Competition.` Cambridge, Mass.: HarvardUniversity Press.

Bank Administration Institute and PSI Global (1997), Profiting From Change in the U.S.Payments System, Bank Administration Institute, Chicago, IL.

Breitkopf, David (2001). ‚First Data Taking Major NYCE Stake,` American Banker, 166 (115),1.

Brenham, Timothy (2001). ‚The Economics of the Microsoft Case,` mimeo.

Cairns, Robert (1994), ‚Asymmetry of Information and Contestability Theory,` Review ofIndustrial Organization, February, 9 (1), 99-107.