به نام خدا

پروژه درس انتقال داده

عنوان پروژه :

ساخت و توسعه یک کتابخانه تشخیص و تصحیح خطا در پایتون

سپند حقیقی

9023076

فهرست

1- مقدمه ..................................................................................................3

2- کلاس اصلی برنامه............................................................................ 4

3- الگوریتم ها ..................................................................................... 7

1-3 بیت توازن ....................................................................................................... 8

2-3 کد های تکرار.................................................................................................... 11

3-3 Checksum....................................................................................................... 12

4-3 CRC..................................................................................................................... 14

5-3 Hamming....................................................................................................... 16

4- تابع های کمکی .....................................................................................18

5- منابع …………......................................................................................19

1- مقدمه :

هدف از این پروژه ساخت و توسعه یک بسته نرم افزاری در زبان برنامه نویسی پایتون است ، از آنجا که امروزه استفاده از ماژول های ارتباط شبکه ای در این زبان برنامه نویسی در حال توسعه است وجود یک کتابخانه برای تشخیص و تصیح خطا در این زبان ضروری به نظر می رسد.

این کتابخانه به صورت دو فایل error\_detect (که شامل کلاس اصلی ) و func(شامل الگوریتم ها) پیاده سازی شده است و مخصوص پایتون 3 می باشد.

آخرین نسخه این کتابخانه را می توانید از آدرس زیر دانلود کنید.

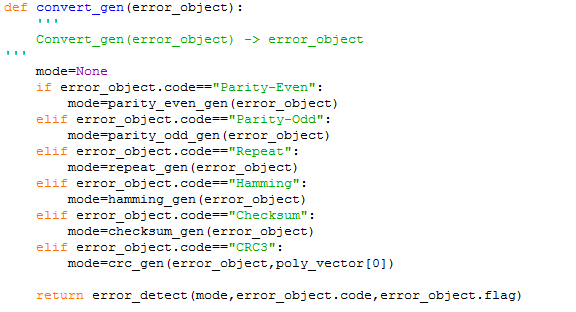
www.shaghighi.ir/python

بهار - 1394

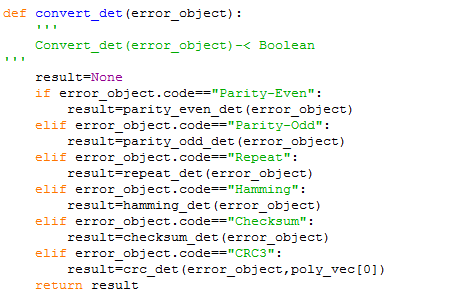
2 – کلاس اصلی برنامه(Error\_Detect) :

در این کلاس شی ای به نام Error\_Detect تعریف شده است که دارای سه پارامتر Code ، Flag و String است که به ترتیب معرف Methodتشخیص خطا ، پارامتر استفاده شده از الگوریتم های تکراری و رشته ارسالی مورد نظر است.

همچنین در این کلاس دو تابع Convert\_gen و Convert\_det تعریف شده است که برای اصلاح و تشخیص رشته ارسالی می باشد که با استفاده از Code شی تابع الگوریتم مورد نظر را از فایل Func فراخوانی کرده و برروی آنها اجرا می کند و همچنین خروجی آن توابع را گرفته و شی Error\_Detect خروجی را تولید می کند.



کد-1 (تابع Convert\_gen)



کد-2 (تابع Convert\_det)

\* همچنین در این کلاس روابط خروجی گرفتن ، طول رشته ، تساوی ، ضرب و جمع برای روی این شی نوشته شده است.

\* مقدار Flag نیز به صورت پیش فرض 2 است.

**3 – الگوریتم ها :**

در نسخه اول این کتابخانه توابع زیر پیاده سازی شده است که در بروزآوری های بعدی افزایش می یابد.

- بیت توازن (Parity Bit)

- کد افزونگی چرخشی (CRC)

- کد های تکراری (Repetition Code)

- Hamming Code

- Checksum

الگوریتم های زیر در حال توسعه اند و در نسخه های بعدی اضافه خواهند شد :

- Adler-32

- Fletcher Checksum

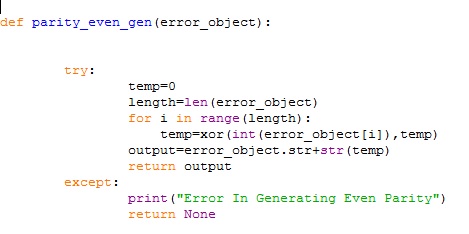
که در ادامه نحوه پیاده سازی هر کدام از الگوریتم ها را به صورت مختصر توضیح داده شده است.

**1-3 بیت توازن (Parity Bit) :**

کد این الگوریتم در این کتابخانه به صورت Parity-Odd و Parity-Even است که به ترتیب نماد بیت توازن فرد و بیت توازن زوج است.

**Parity\_even\_gen :**

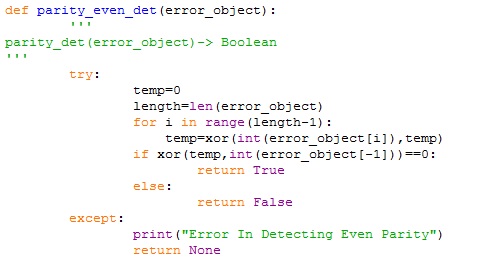
این تابع یک شی از نوع Error\_Detect را دریافت کرده ، رشته پیام آن را که به صورت بیتی است جدا کرده و بیتی برابر XOR سایر بیت ها را به انتهای رشته اضافه کرده و رشته اصلاح شده را به عنوان خروجی بر می گرداند.



کد-3 (تابع parity\_even\_gen)

**Parity\_even\_det :**

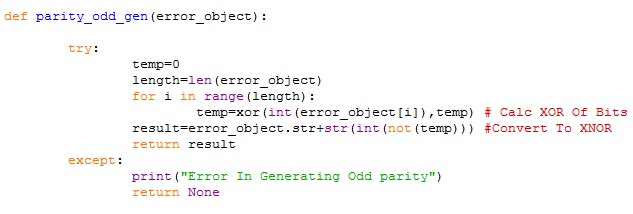
این تابع یک شی از نوع Error\_Detectرا دریافت کرده ، رشته پیام آن را که به صورت بیتی است جدا می کند و تمام بیت های آن را با هم XOR می کند اگر نتیجه صفر شود مقدار منطقی True را بر می گرداند در غیر این صورت False.



کد-4 (تابع parity\_even\_det)

**Parity\_odd\_gen :**

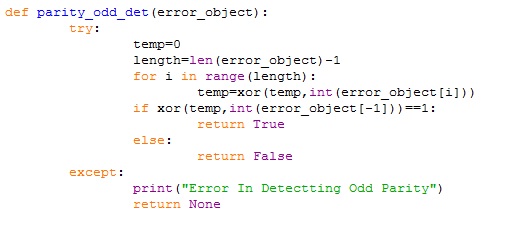
این تابع یک شی از نوع Error\_Detect را دریافت کرده ، رشته پیام آن را که به صورت بیتی است جدا کرده و بیتی برابر XNOR سایر بیت ها را به انتهای رشته اضافه کرده رشته اصلاح شده را به عنوان خروجی بر می گرداند.



کد-5 (تابع parity\_odd\_gen)

**Parity\_odd\_det :**

این تابع یک شی از نوع Error\_Detectرا دریافت کرده ، رشته پیام آن را که به صورت بیتی است جدا می کند و تمام بیت های آن را با هم XOR می کند اگر نتیجه یک شود مقدار منطقی True را بر می گرداند در غیر این صورت False.



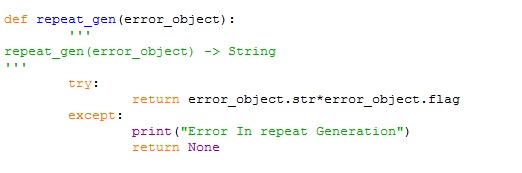
کد-6 (تابع parity\_odd\_det)

**2- 3کد های تکراری** :

کد این الگوریتم به صورت دو تابع Gen و Det برای ساخت و تشخیص کد خطا پیاده سازی شده است.

**Repeat\_gen :**

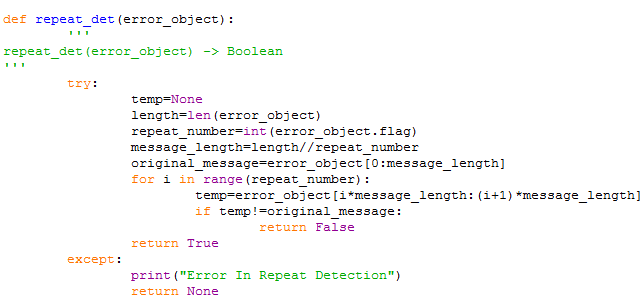
این تابع یک شی از نوع Error\_Detect را دریافت کرده ، رشته پیام آن را که به صورت بیتی است جدا کرده و با توجه به مقدار پارامتر Flag در شی پیام را تکرار کرده و به ادامه رشته می چسباند و در انتها رشته اصلاح شده را به عنوان خروجی باز می گرداند.



کد-7 (تابع repeat\_gen)

**Repeat\_det :**

این تابع یک شی از نوع Error\_Detect را دریافت کرده ، رشته پیام آن را که به صورت بیتی است جدا کرده و با توجه به مقدار پارامتر Flag در شی ، پیام را اصلی را با پیام های تکرار شده مقایسه می کند در صورتی که تفاوتی وجود داشته باشد مقدار منطقی False و در غیر این صورت True رابر می می گرداند.



کد-8 (تابع repeat\_det)

**:Checksum3-3**

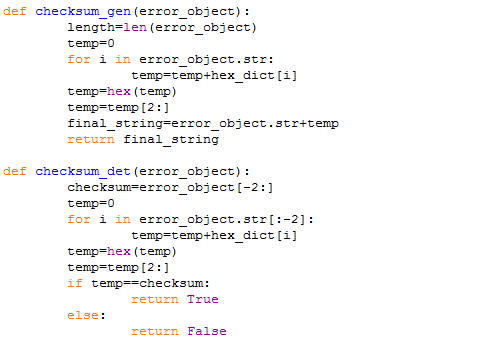
کد این الگوریتم به صورت دو تابع Gen و Det برای ساخت و تشخیص کد خطا پیاده سازی شده است.

**Checksum\_gen :**

این تابع یک شی از نوع Error\_Detect را دریافت کرده ، رشته پیام آن را که به صورت بسته هگز است جدا کرده ، و جمع اعضای آن را به صورت هگز به انتهای آن اضافه می کند و رشته اصلاح شده را بر می گرداند.

**Checksum\_det :**

این تابع یک شی از نوع Error\_Detect را دریافت کرده ، رشته پیام آن را که به صورت بسته هگز است جدا کرده و جمع ارقام آن ( به غیر از دو رقم آخر) را به صورت هگز محاسبه می کند در صورتی که این جمع با دو رقم آخر مساوی باشد مقدار منطقی Trueو در غیر این صورت مقدار منطقی False را بر می گرداند.



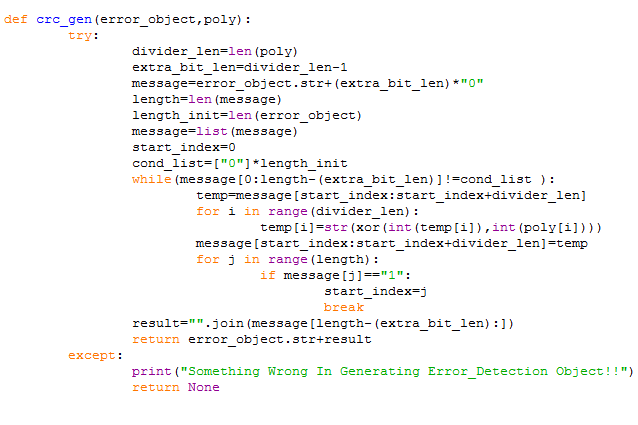
کد-9 (توابع checksum)

4-3 کد افزونگی چرخشی (CRC) :

کد این الگوریتم به صورت دو تابع Gen و Det برای ساخت و تشخیص کد خطا پیاده سازی شده است.

**Crc\_gen :**

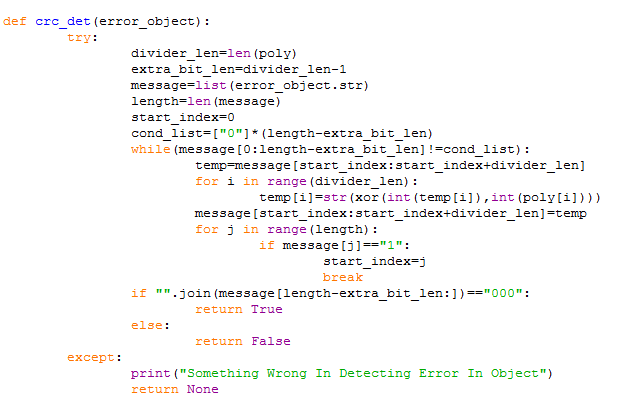
این تابع یک شی از نوع Error\_Detect و ضرایب چند جمله ای را به عنوان ورودی گرفته ، رشته پیام شی را که به صورت بیتی بوده جداکرده به اندازه درجه چند جمله ای به انتهای صفر اضافه کرده و چند جمله ای را از چپ برروی آن حرکت داده و بیت به بیت XORمی کند و در انتهای الگوریتم نیز رشته اصلاح شده را در خروجی قرار می دهد.



کد-10(تابع crc\_gen)

**Crc\_det :**

این تابع نیز یک شی از نوع Error\_Detect و ضرایب چند جمله ای را به عنوان ورودی گرفته و رشته پیام شی را که به صورت بیتی بوده جدا می کند. در ادامه چند جمله ای را برروی رشته حرکت داده و بیت به بیت XOR می کند و باقی مانده نهایی را حساب می کند در صورتی که باقی مانده صفر باشد مقدار منطقی Trueو در غیر این صورت False را بر می گرداند.

****

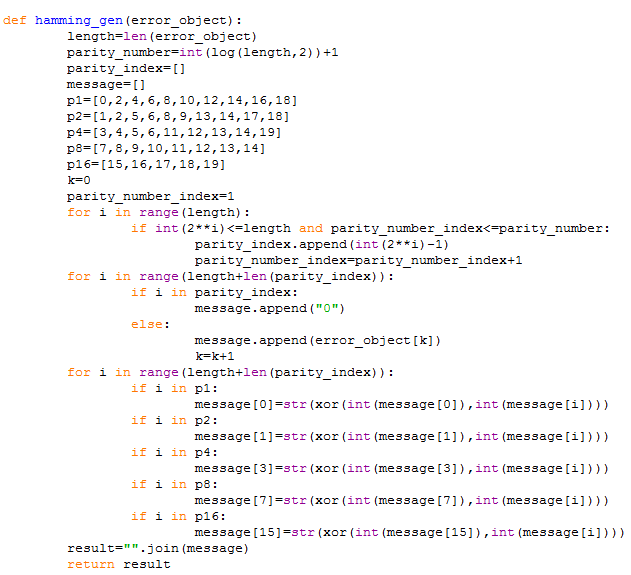
کد-11(تابع crc\_det)

**:Hamming Code5-3**

کد این الگوریتم به صورت دو تابع Gen و Det برای ساخت و تشخیص کد خطا پیاده سازی شده است.

**Hamming\_gen :**

این تابع یک شی از نوع Error\_Detect را دریافت کرده ، رشته پیام آن را که به صورت بیتی است جدا کرده و با توجه به طول رشته بیت های توازن را به آن اضافه کرده و رشته خام جدید را تشکیل می دهد در ادامه با XOR کردن بیت های مربوط به هر کدام از بیت های توازن رشته جدید را ایجاد کرده و در خروجی قرار می دهد.



کد-12(تابع hamming\_gen)

**Hamming\_det :**

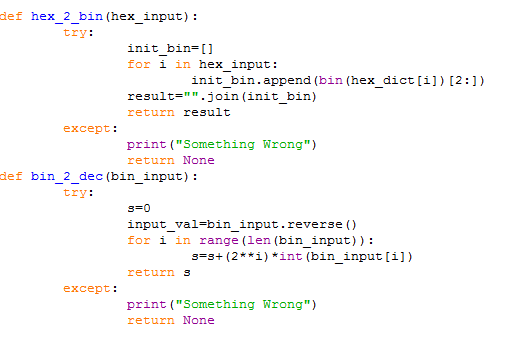
این تابع یک شی از نوع Error\_Detect را دریافت کرده ، رشته پیام آن را که به صورت بیتی است جدا کرده و با توجه به طول رشته بیت های توازن را به آن را تشخیص می ، در ادامه با XOR کردن بیت های مربوط به هر کدام از بیت های توازن با خود بیت توازن ، وجود یک بیت خطا و مکان خطا و یا عدم وجود خطا را اعلام می کند.



کد-13(تابع hamming\_det)

**4 – توابع کمکی :**

برای سهولت در استفاده از این کتابخانه توابع تبدیلی بین باینری ، دسیمال و هگز نوشته شده است تا به راحتی بتوان رشته های مورد نظر را به هم تبدیل کرد.

****

کد-14(توابع تبدیلی کمکی)

**5 – منابع :**

[1] Shu Lin, Daniel J. Costello, Jr. (1983). *Error Control Coding: Fundamentals and Applications*

[2] Wilson R, Lammers D. (2004). Soft errors become hard truth for logic EE Times Online. May 3 2004. <http://www.eetimes.com/showArticle.jhtml?articleID=19400052>

[3] “Error Detection And Correction.” Wikipedia :The Free Encyclopedia. Wikimedia Foundation , <http://en.wikipedia.org/wiki/Error_detection_and_correction>

[4] “Cyclic Redundancy Check.” Wikipedia :The Free Encyclopedia. Wikimedia Foundation , <http://en.wikipedia.org/wiki/Cyclic_redundancy_check>

[5] “Hamming Code.” Wikipedia :The Free Encyclopedia. Wikimedia Foundation , <http://en.wikipedia.org/wiki/Hamming_code>