Project On

Mathematical Captcha



Prepared by : Aditya Sarkar(B-52)

Ankit Sharma(B-53)

Amanjot Singh(B-54)

Baldev Sandhu(B-55)

**Objective**

The objective of this project is to make a mathematical based captcha using python. Captcha is a challenge –response system to protect website access against bots. CAPTCHA prevents spam in website comment sections and on blogs. Many spammers bombard comment sections with links to increase search engine rankings. The test makes sure only humans comment and users don’t have to sign in beforehand to leave a comment. Many companies offer free email services but a while ago bots would sign up for hundreds of free accounts and then use these accounts to cause havoc on the Internet. Now people need to complete a CAPTCHA before being able to get a free email account. Free services should be protected by CAPTCHA to prevent abuse via automated scripts. It offers protection from scrapers who want to copy the email addresses of users. Spammers would crawl the Internet for email addresses that are posted in clear text. By utilising CAPTCHA you can protect against these scrapers. People need to solve a CAPTCHA before an email address is shown. Sometimes people don’t want a webpage to be shown so there is an HTML tag that hides the page from robots. Big search engine companies respect this but sometimes it doesn’t prevent all bots from coming through. This is what CAPTCHA helps to prevent. Dictionary attacks are when a computer goes through every word in a dictionary in order to obtain access to someone’s password and account. CAPTCHAs prevents this by requiring the computer (or person) to enter a code after a certain amount of unsuccessful logins.

**Introduction**

A CAPTCHA an acronym for "Completely Automated Public Turing test to tell Computers and Humans Apart" is a type of challenge–response test used in computing to determine whether or not the user is human. The term was coined in 2003 by Luis von Ahn, Manuel Blum, Nicholas J. Hopper, and John Langford. Quite a mouthful. It was coined in 2000 by professors and scientists from Carnegie Mellon University and IBM. The most common type of CAPTCHA (displayed as Version 1.0) was first invented in 1997 by two groups working in parallel. The first group consisted of Mark D. CAPTCHA is a difficult acronym but is a very useful tool in differentiating robots from humans. In short, it is a simple test to detemine if the user is a robot or human. A CAPTCHA is what is called a challenge-response test. One party presents a question or challenge and the other party must provide a valid answer or response in order to be authenticated.

The CAPTCHA idea originally comes from the Turing test (as can be seen in the above acronym). A Turing test is a means with which to test a machine’s ability to exhibit intelligent behaviour equivalent to that of a human being. A CAPTCHA can be called a reverse Turing test since it is a computer creating the test in the first place that will challenge humans and not the other way around. CAPTCHA prevents spam in website comment sections and on blogs. Many spammers bombard comment sections with links to increase search engine rankings. The test makes sure only humans comment and users don’t have to sign in beforehand to leave a comment.

Many companies offer free email services but a while ago bots would sign up for hundreds of free accounts and then use these accounts to cause havoc on the Internet. Now people need to complete a CAPTCHA before being able to get a free email account. Free services should be protected by CAPTCHA to prevent abuse via automated scripts. It offers protection from scrapers who want to copy the email addresses of users. Spammers would crawl the Internet for email addresses that are posted in clear text. By utilising CAPTCHA you can protect against these scrapers. People need to solve a CAPTCHA before an email address is shown.

Sometimes people don’t want a webpage to be shown so there is an HTML tag that hides the page from robots. Big search engine companies respect this but sometimes it doesn’t prevent all bots from coming through. This is what CAPTCHA helps to prevent. Often people will use programs to stuff online polls in favour of a certain vote. Usually IP addresses are recorded to prevent people from voting more than once but with the use of bots one can circumvent this policy. This makes it hard to truly trust online polls if CAPTCHA codes are not involved. Dictionary attacks are when a computer goes through every word in a dictionary in order to obtain access to someone’s password and account. CAPTCHAs prevents this by requiring the computer (or person) to enter a code after a certain amount of unsuccessful logins. The prevention of torrent sites from bots falsifying seed counts and positive reviews in order to trick people into download a trojan virus.

Now-a-days, websites became the identity for many businesses. Many companies have their sites for online business. These companies offer a lot of free services to their users. The only thing is to register on the site.

So, many people exploit the offered free services by duplicate registration. They write a computer program that can automatically register on the site and then can use the offered services. CAPTCHA is developed for avoiding such spam registrations on the site and exploitation of the offered free services.

CAPTCHA is a distorted image containing short text. It is displayed in such a format so that only human eyes can recognise the alphabets clearly. At the time of registration, such image is displayed on the form and the user is asked to write the same text in given text field. The robots fail to recognise the short text. Thus, website owners can prevent robots from registration and can ensure that all the members using free services are humans.

Thus CAPTCHAs prevent automated posting to blogs and forums. CAPTCHAs can be used further in avoiding spam emails. Apart from these usage of CAPTCHAs; it is criticised that people with poor eyesight or blind people will be unable to use the web services offered.

The need for CAPTCHAs began as far back as 1997. At that time, the internet search engine AltaVista was looking for a way to block automated URL submissions to the platform which were skewing the search engine's ranking algorithms. To solve the problem, Andrei Broder, AltaVista's chief scientist, developed an algorithm that randomly generated an image of printed text. Although computers couldn't recognize the image, humans could read the message the image contained and respond appropriately. Broder and his team were issued a patent for the technology in April 2001. In 2003, Nicholas Hopper, Manuel Blum, Luis von Ahn of Carnegie Mellon University and John Langford of IBM perfected the algorithm and coined the term

Types of CAPTCHAS

The most common type of CAPTCH is the text CAPTCHA, which requires the user to view a distorted string of alphanumeric characters in an image and enter the characters in an attached form. Text CAPTCHAS are also rendered as MP3 audio recordings to meet the needs of the visually impaired. Just as with images, bots can detect the presence of an audio file, but only a human can listen and know the information the file contains.

Picture recognition CAPTCHAs, which are also commonly used, ask users to identify a subset of images within a larger set of images. For instance, the user may be given a set of images and asked to click on all the ones that have cars in them.

Other types of CAPTCHAs include:

Math CAPTCHAs - require the user to solve a basic math problem, such as adding or subtracting two numbers.

3D Super CAPTCHAs - require the user to identify an image rendered in 3D.

I am not a robot CAPTCHA - requires the user to check a box.

Marketing CAPTCHAs - require the user to type a particular word or phrase related to the sponsor's brand.

Bypassing CAPTCHA-

Users who prefer not to solve CAPTCHAs can use any of several browser add-ons that allow users to bypass CAPTCHAs. Popular browser add-ons include AntiCapture, CAPTCHA Be Gone and Rumola.

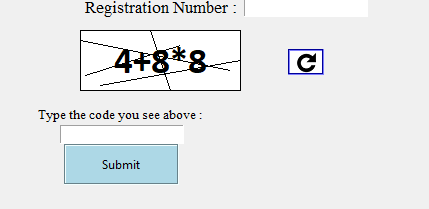
The AntiCaptcha automatic CAPTCHA solver plug-in for Chrome and Firefox automatically finds a CAPTCHA on a webpage and solves it for the user. This extension is promoted as being helpful for users with vision impairments, as well as for users who prefer to bypass CAPTCHA codes. As of this writing, the cost of the service starts at $0.70 for 1,000 CAPTCHA images. The CAPTCHA Be Gone extension detects CAPTCHAs on webpages, solves them and copies the result to a user's clipboard. At this time, the utility is available for Firefox, Chrome and Internet Explorer for a $3.50 per month subscription fee. The Rumola add-on for Firefox, Chrome and Safari automatically searches for CAPTCHAs on the webpages a user visits. Currently, the cost is either $0.95 for 50 CAPTCHA solutions or $1.95 for 150 solutions. There is also a JavaScript bookmarklet that can be used for devices that connect to the internet. Because CAPTCHA bypass add-ons are created by third parties, end users should be aware that the browser extension could expose their browsing activity to untrusted sources. Another reason not to use CAPTCHA bypasses is that the performance of the extensions is inconsistent. This is primarily because as bots get smarter, CAPTCHAs are also evolving and it can be difficult for the add-on programs to keep up.

**Source Code**

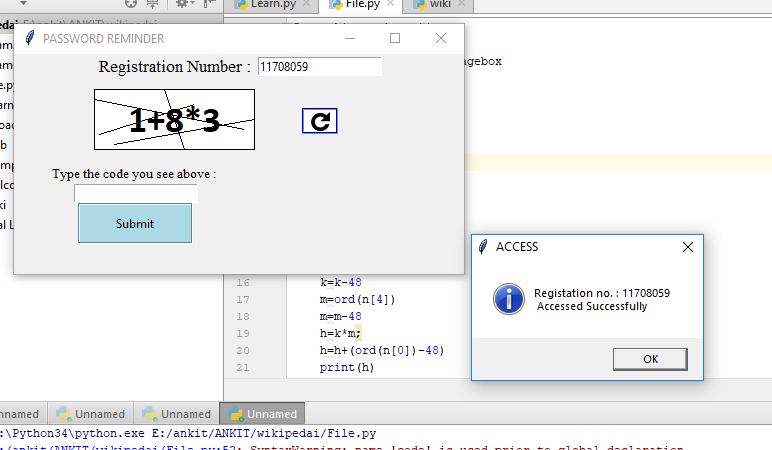
**from** tkinter **import**\*  
  
**from** tkinter **import** messagebox  
  
**import** tkinter  
  
**import** random  
  
top=tkinter.Tk()  
  
code=**''  
  
  
def** eval(n):  
 k=ord(n[2])  
 k=k-48  
 m=ord(n[4])  
 m=m-48  
 h=k\*m;  
 h=h+(ord(n[0])-48)  
 print(h)  
 **return** h  
  
  
**def** gen\_cap():  
  
 n=**''** result=**''  
 for** i **in** range(0,6):  
 **if** i==0:  
 value=random.randint(48,57)  
 n += chr(value)  
 **if** i==1:  
 n=n+**"+"  
 if** i==2:  
 value = random.randint(48, 57)  
 n += chr(value)  
 **if** i==3:  
 n=n+**"\*"  
 if** i==4:  
 value = random.randint(48, 57)  
 n += chr(value)  
  
 **return** n  
  
**def** check():  
 result = eval(code)  
 ck=ent\_cap.get()  
  
 ent\_cap.delete(0,END)  
  
 **global** code  
  
 r=Reg\_ent.get()  
  
 **if**(r.isdigit() **and** (len(r)==8 **or** len(r)==5)):  
  
 Rgn=Reg\_ent.get()  
  
 **else**:  
  
 messagebox.showinfo(**"ERROR"**,**"Reg no. Not valid"**)  
  
 Reg\_ent.delete(0,END)  
  
 **return** 0  
 ck=str(ck)  
 result=str(result)  
 **if**(ck==result):  
 messagebox.showinfo(**"ACCESS"**,**"Registation no. : %s \n Accessed Successfully"**%Rgn)  
 **else** :  
  
 messagebox.showinfo(**"ERROR"**,**"Wrong captcha"**)  
  
 g=gen\_cap()  
  
 code=g  
  
 display()  
  
 c.create\_text(160,40,text=g,font=**'Calibri 28 bold'**)  
  
 c.grid(row=3,column=10)  
  
 see=tkinter.Label(top,text=**'re-enter : '**,font=**'Times 10'**)  
  
 see.grid(row=6,column=10)  
  
**def** display():  
  
 c.create\_rectangle(80,10,240,70,fill=**'white'**)  
  
 c.create\_line(80,20,230,50)  
  
 c.create\_line(85,55,180,25)  
  
 c.create\_line(150,10,170,70)  
  
 c.create\_line(100,65,240,40)  
  
  
  
**def** reset1():  
  
 ent\_cap.delete(0,END)  
  
 **global** code  
  
 g=gen\_cap()  
  
 code=g  
  
 display()  
  
 c.create\_text(160,40,text=g,font=**'Calibri 28 bold'**)  
  
 c.grid(row=3,column=10)  
  
code=gen\_cap()  
Reg=tkinter.Label(top,text=**'Registration Number : '**,font=**'Times 13'**)  
  
Reg.grid(row=1,column=10,sticky=E)  
  
Reg\_ent=Entry(top)  
  
Reg\_ent.grid(row=1,column=11)  
  
top.geometry(**'450x220'**)  
  
top.title(**"PASSWORD REMINDER"**)  
  
c=Canvas(top,height=80,width=240)  
  
display()  
  
c.create\_text(160,40,text=code,font=**'Calibri 28 bold'**)  
  
c.grid(row=3,column=10)  
  
see=tkinter.Label(top,text=**'Type the code you see above : '**,font=**'Times 10'**)  
  
see.grid(row=4,column=10)  
  
ent\_cap=Entry(top)  
  
ent\_cap.grid(row=5,column=10)  
  
submit=tkinter.Button(top,text=**'Submit'**,relief=GROOVE,command=check,height=2,width=15,bg=**'lightblue'**)  
  
submit.grid(row=7,column=10)  
  
img=PhotoImage(file=**"reload.png"**)  
  
reset=tkinter.Button(top,text=**'Reload'**,relief=GROOVE,height=20,width=30,bg=**'blue'**,image=img,command=reset1)  
  
reset.grid(row=3,column=11)  
  
top.mainloop()

**Screenshots of GUI**

**#1**

****

**#2**

****

**Result**

The result of this project is to verify the user is a human or bots. Here when the user write the correct captcha that is written in the box then it gets verified, and the user can get access to next page. But if the user write wrong captcha whether by mistake or anything else then a dialog box will show to remind the user that you have entered the wrong captcha , and give the chance to user to correct it. The user is not allowed to process to next page until and unless he/she writes the correct captcha in the box and get verified that the user is human.

**Refrences**

1. www.youtube.com
2. www.quora.com
3. www.stackoverflow.com
4. www.wikipedia.com
5. www.maxxor.com