**Function:** toupper()

**Definition:** It changes a lowercase value into an uppercase one. I don’t see a lot of use for this unless one was dealing with more GUI related programs and it could be altered to alternate lowercase and uppercase in a loading menu. Like when Metasploit starts up.

**Code:**

#include <stdio.h>

#include <ctype.h>

int main () {

int i = 0;

char str[] = "Tutorials Point";

while(str[i]) {

putchar (toupper(str[i]));

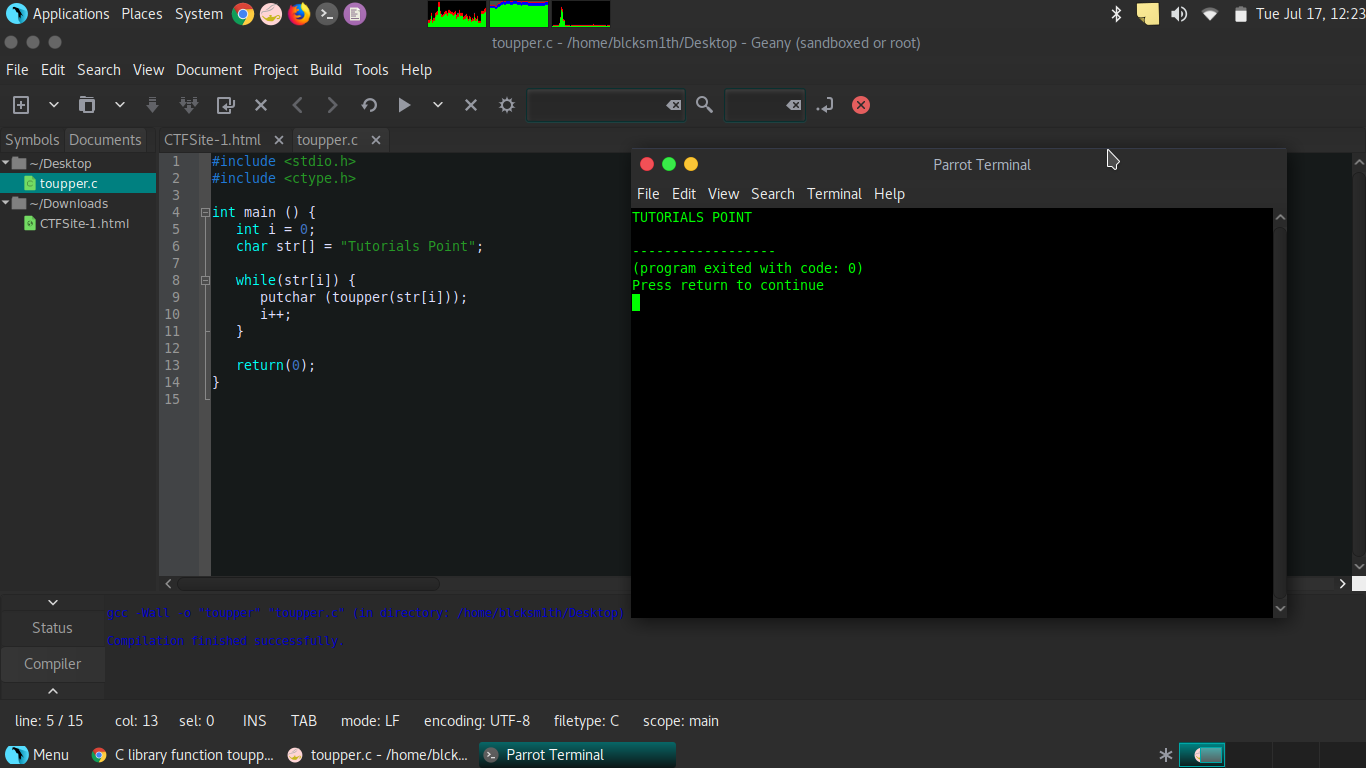
i++;

}

return(0);

}

**Snapshot:**



**Function:** isdigit()

**Definition:** Determines if a value is a integer between 0-9. If you are determining what is considered a digit it could be used. One could also put a loop within a loop and make the program do extra functions if the value is a digit, verses if its not. It’s mathematic in nature so it can be used mathematically to create algorithms.

**Code:**

#include <stdio.h>

#include <ctype.h>

int main()

{

char c;

printf("Enter a character: ");

scanf("%c",&c);

if (isdigit(c) == 0)

printf("%c is not a digit.",c);// every other character

else

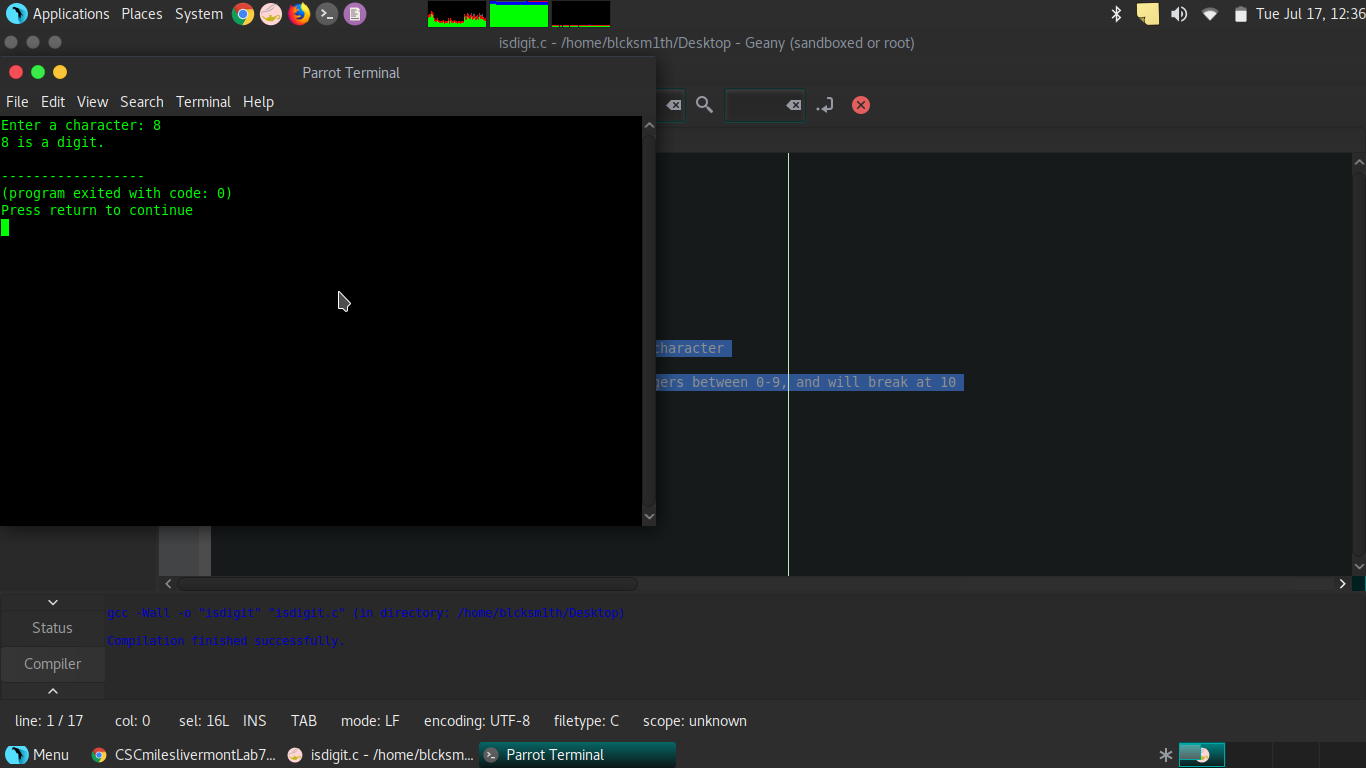
printf("%c is a digit.",c); //only counts integers between 0-9, and will break at 10

return 0;

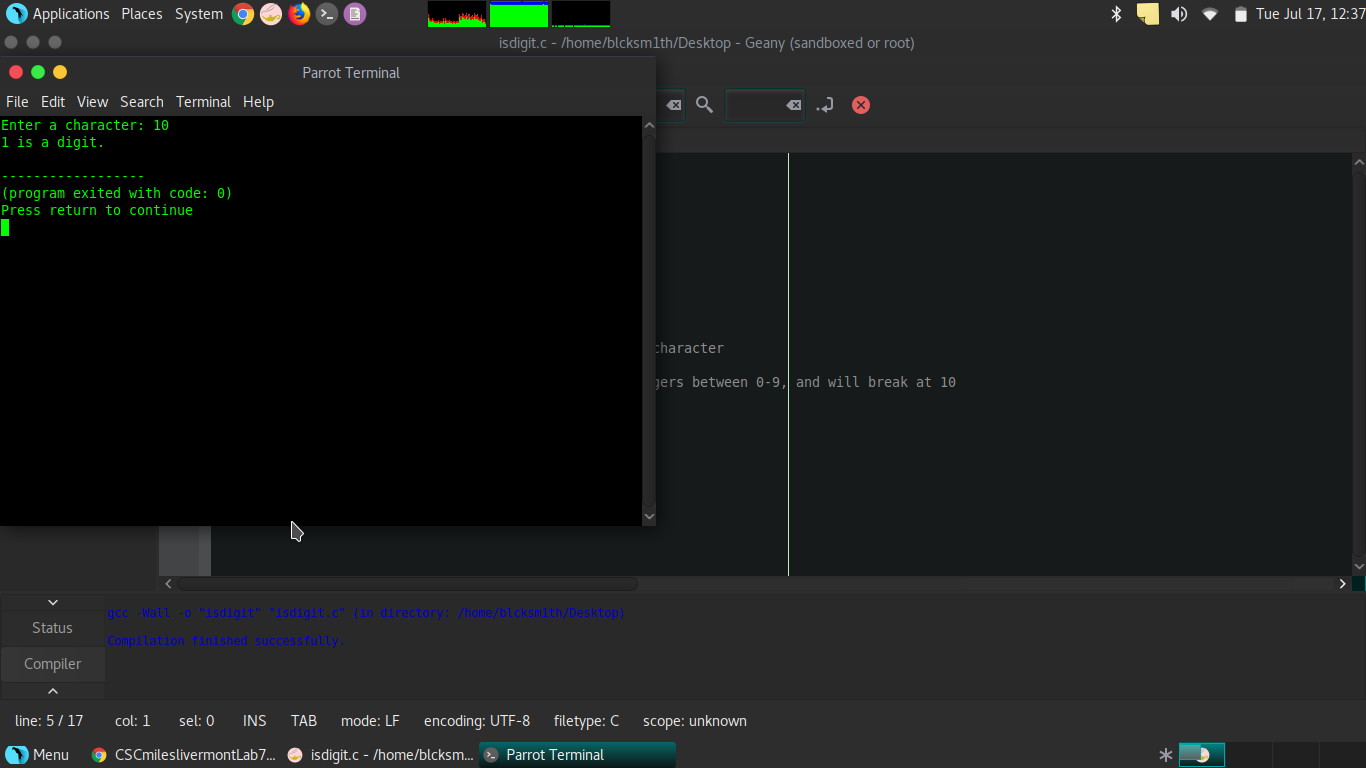
}

**Snapshots:**

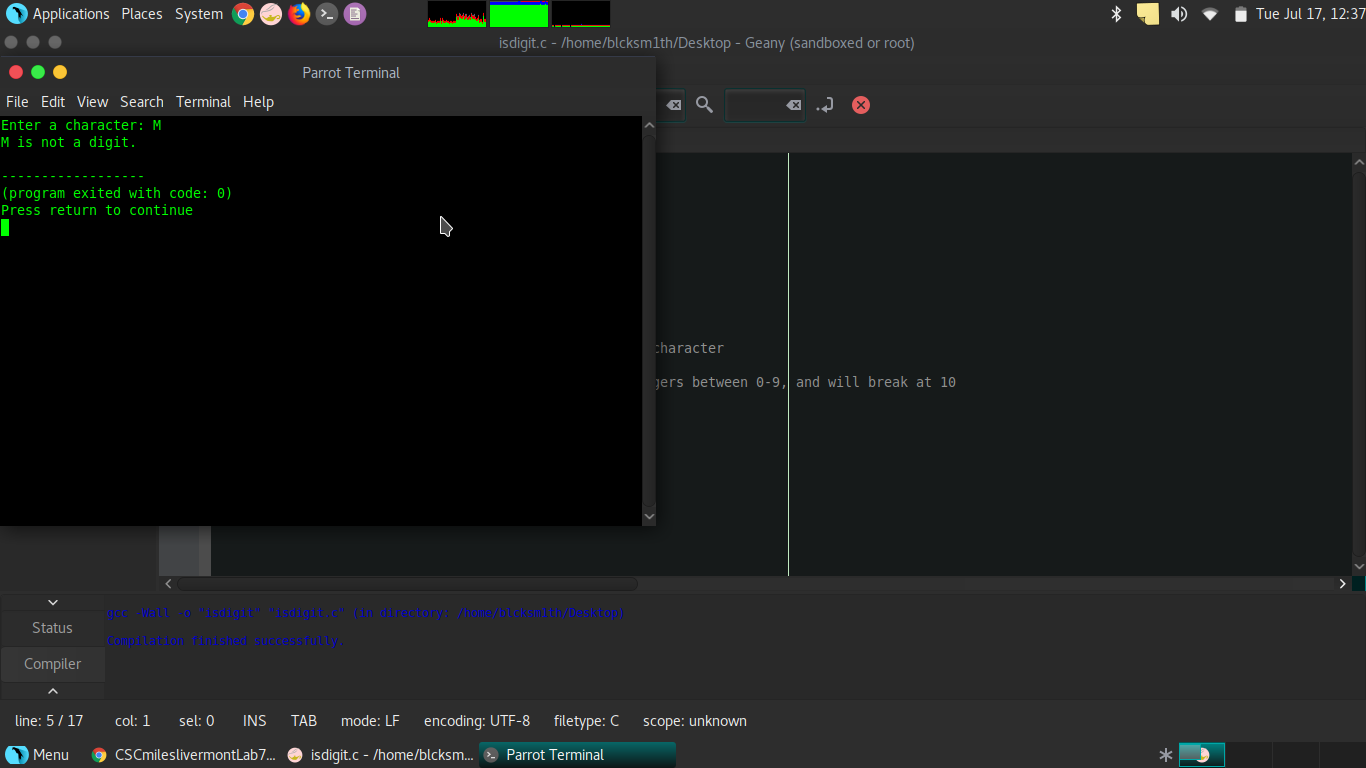
First shot digit 8:



Second shot digit 10:



Third shot letter M:



**Function:** strlen()

**Definition:** Finds the length of string of characters, before the terminating value. This can be used as another way to handle user input. Example, if the question is, is Monty Python and the Holy Grail; epic or amazing, one can create multiple outputs based on the answer. Such as if string == 4, then output is puts (“I think you're absolutely right”), but else the output is puts(“ I agree completely. “). It is a way to convert words into numbers the computer can use.

**Code:**

#include <stdio.h>

#include <string.h>

int main()

{

char c[20];

printf("Enter string: ");

gets(c);

printf("Length of string c = %d \n",strlen(c));

if ( strlen(c) == 5 )

{

puts("Good morning Mr.Stark");

}

else

{

puts("My name is Vision now, not Jarvis");

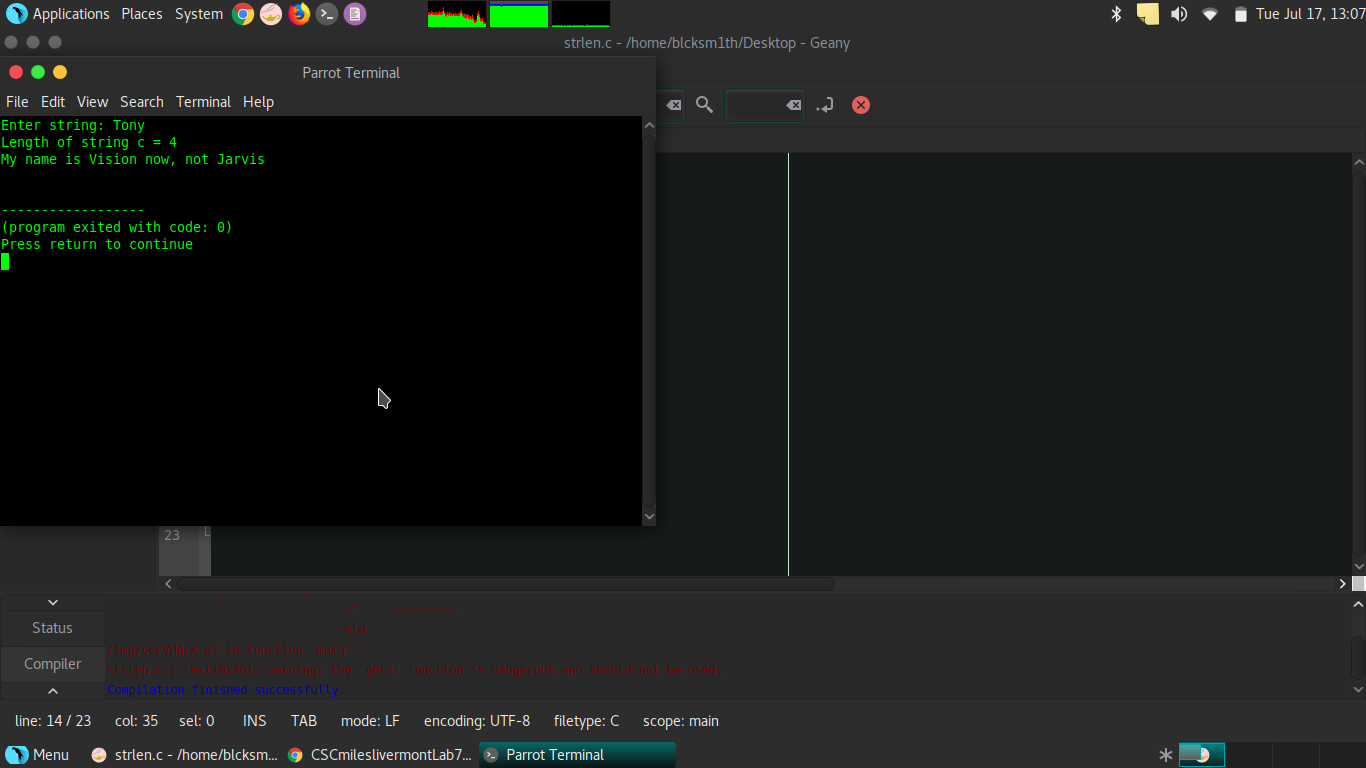
}

return 0;

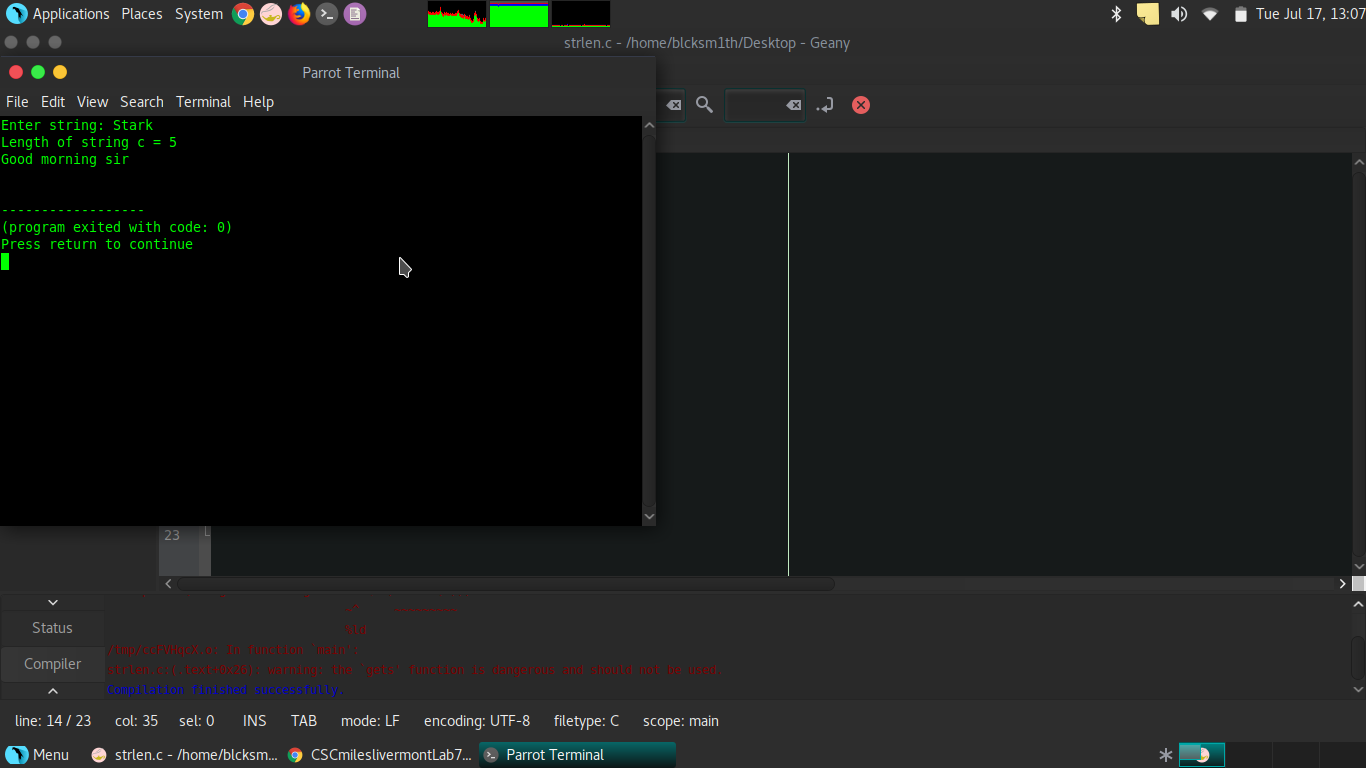
}

**Snapshots:**

Snapshot of Tony;



Snapshot of Stark;



**Function:** strcat()

**Definition:** It takes two string and puts them together. Like the one game where one person says a word and the other responds. Example, Fortnite is... ,... overrated. strcat() takes words and stings and puts them together. So if the input is ‘Fortnite is” and ‘overrated’, then the output would be ‘Fortnite is overrated’. Besides guessing games and insults it can be used as the basis for a program to respond to a user, and finish their sentences. Like when google tries to autofill when one is using their search engine, but that is more likely written in a JavaScript equivalent.

**Code:**

#include <stdio.h>

#include <string.h>

int main()

{

char str1[] = "Fortnite is", str2[] = "overrated!";

strcat(str1,str2);

puts("Fornite is...");

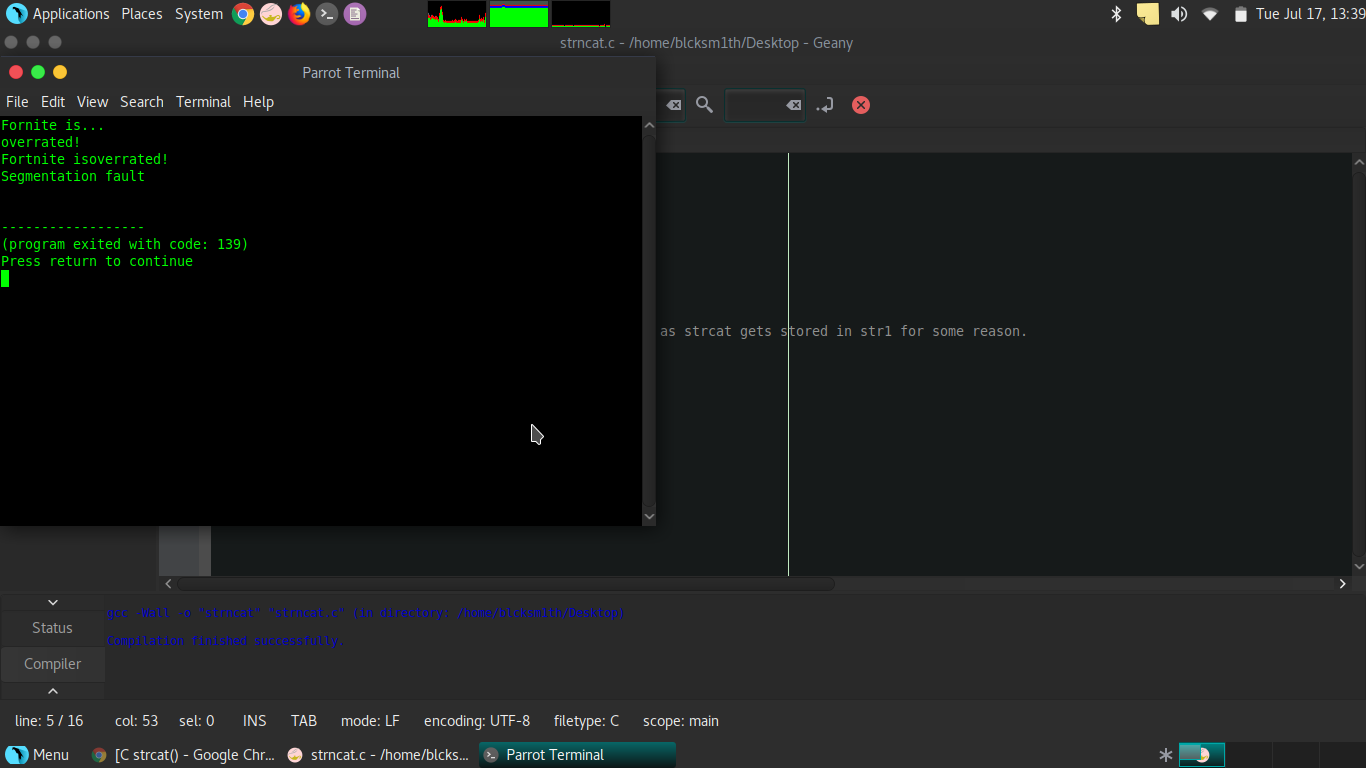
puts("overrated!");

puts(str1); // this will show to combined input, as strcat gets stored in str1 for some reason.

return 0;

}

**Snapshot:**

****

**Function:** strchr()

**Definition:** Acts like a mini-search engine within the program and can look for specific characters and can makes it easier to create algorithms that manipulate input. For example if someone had a string that says “I love to eat, grandma”, it can be used to search for ‘,’, and then one can right a manipulated printf function to give the output, “grandma”. If applied to more complex ideas though if this had abilities to be modified to search for common patterns in an input it has possible decryption possibilities with this function.

**Code:**

#include <stdio.h>

#include <string.h>

int main () {

const char str[] = "<http://www.montypython3.com>"; //this is an actual site I have started developing for my CTF team Montypython3

const char ch = '.';

char \*ret;

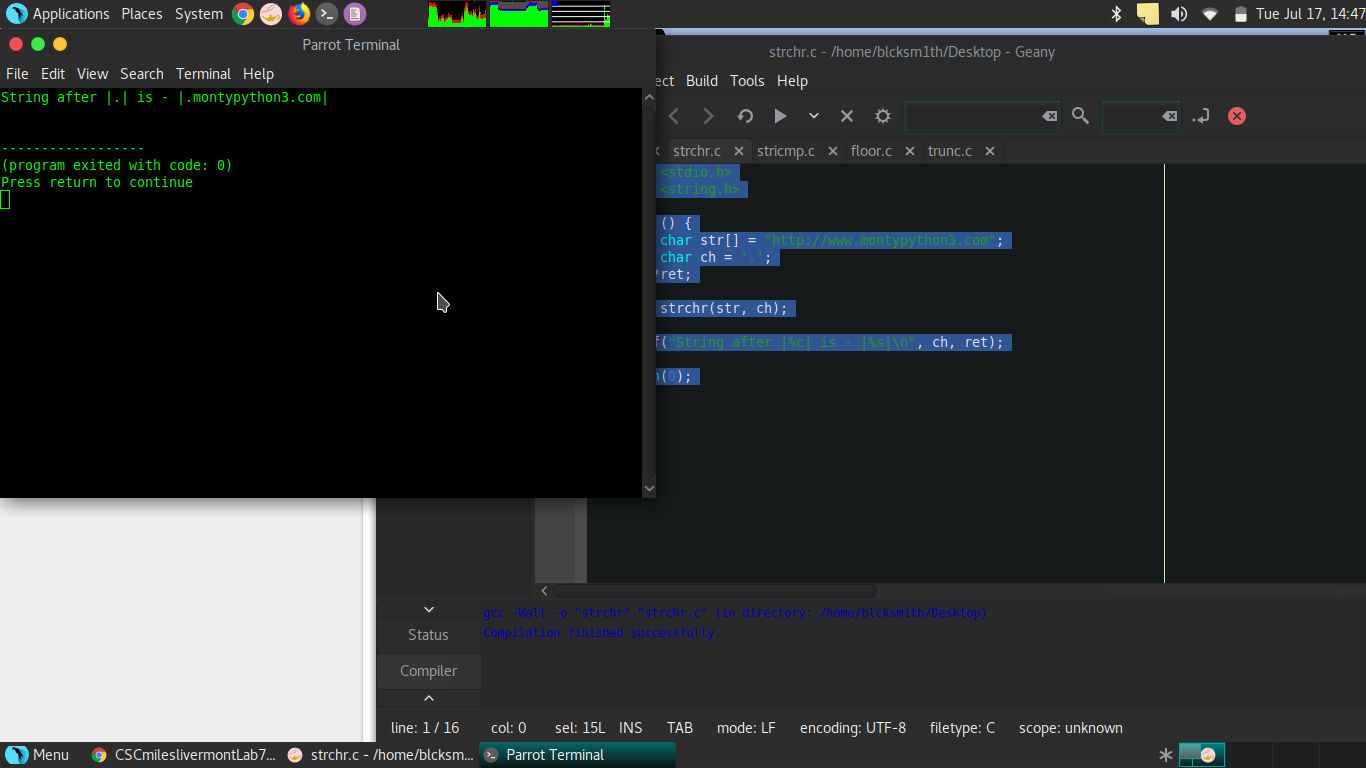
ret = strchr(str, ch);

printf("String after |%c| is - |%s|\n", ch, ret);

return(0);

}

**Snapshot:**

****

**Function:** stricmp(), strcasecmp in Unix/Linux

**Definition:** This is used to compare different strings to find commonalities. Example, str1 = ‘knock, knock Penny’, str2 = ‘knock, knock, Penny’, str3 = ‘knock, knock Penny’ , what can right a function that can determine if the strings are identical or not. In this case, ‘knock, knock Penny’, ‘knock, knock Penny’, and ‘knock, knock Penny’ are all identical. Including A= a, unlike strcmp. The code produces a 0 when they all match and a positive integer when the strings don’t. This one has comparative analysis applications and could be used to find commonalities in stock statistics or to write a program that can see if students are plagiarizing.

**Code:**

#include <string.h>

#include <stdio.h>

int main()

{

char str1[] = "abcd", str2[] = "abCd", str3[] = "abcd";

int result;

// comparing strings str1 and str2

result = strcasecmp(str1, str2);

printf("strcasecmp(str1, str2) = %d\n", result);

// comparing strings str1 and str3

result = strcasecmp(str1, str3);

printf("strcasecmp(str1, str3) = %d\n", result);

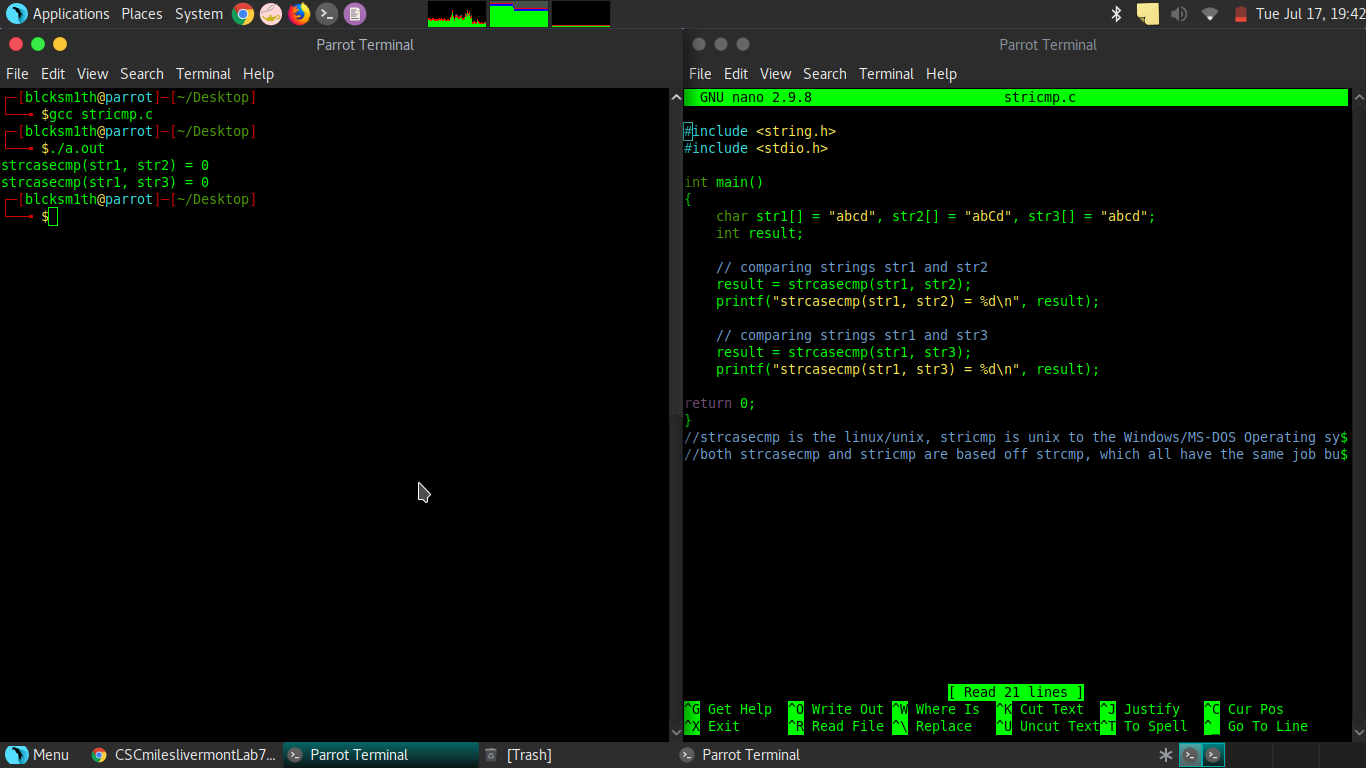
return 0;

}

//strcasecmp is the linux/unix, stricmp is unix to the Windows/MS-DOS Operating systems

//both strcasecmp and stricmp are based off strcmp, which all have the same job but unlike strcmp the other two are not case sensitive. A=a

**Snapshot:**



**Function:** floor()

**Definition:** Floor() is used to find the largest integer that can into a number, based on an argument. Example, for the number 10.489437843947, the largest integer that can go into this is 10. This has more application to mathematics or science where significant digits and estimation is important. Especially when measuring and calculating input from instruments that have limited accuracy, such as a thermometer.

**Code:**

#include <stdio.h>

#include <math.h>

int main()

{

float i=4.20, j=6.9, k=-9.11, l=-3.1;

printf("floor of %f is %f\n", i, floor(i));

printf("floor of %f is %f\n", j, floor(j));

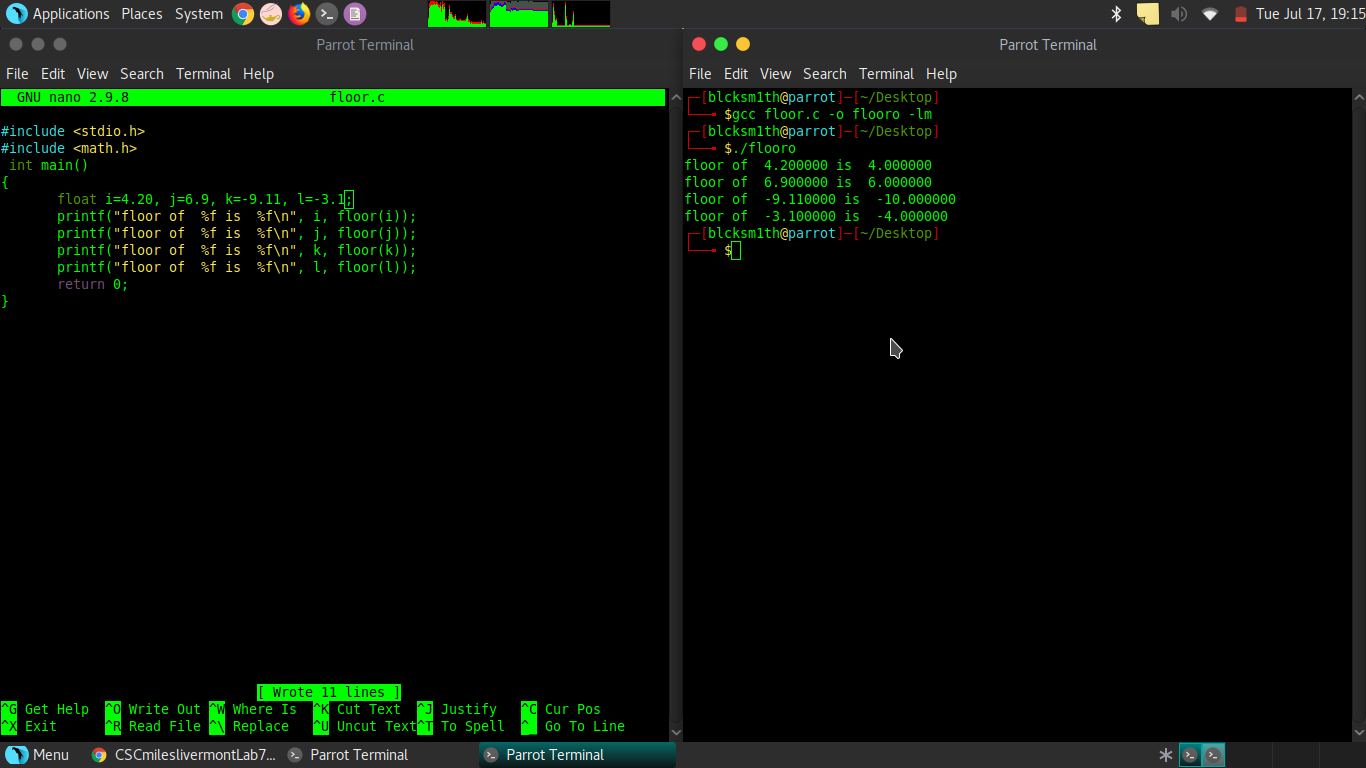
printf("floor of %f is %f\n", k, floor(k));

printf("floor of %f is %f\n", l, floor(l));

return 0;

}

**Snapshot:**



**Function:** trunc()

**Definition:** Is used to modify integers and decimals, and give a returned result. This is like other functions that are used to manipulate information for the computer to create algorithms. Specifically it truncates the decimal value from floating point value and returns integer value. It is used to return values like date and time.

**Code:**

#include <stdio.h>

#include <math.h>

int main()

{

printf ("truncated value of 16.99 = %f\n", trunc (16.99) );

printf ("truncated value of 20.1 = %f\n", trunc (20.1) );

return 0;

}

**Snapshot:**

