**Bandit OverTheWire.org upto level 15**

**Bandit0**

[\*] Used ssh cmd to login

**ssh bandit0@bandit.labs.overthewire.org -p 2220**

username : bandit0,

Password: bandit0

**Bandit0 -> Bandit1**

[\*] Used cat cmd to read

**cat readme**

username : bandit1

Password: boJ9jbbUNNfktd78OOpsqOltutMc3MY1

**Bandit1 -> Bandit2**

Password is stored in a - file

[\*] Used cat cmd followed by ./- to read

**cat ./-**

username : bandit2

Password: CV1DtqXWVFXTvM2F0k09SHz0YwRINYA9

**Bandit2 -> Bandit3**

Password is stored in spaces in this filename

[\*] Used cat cmd followed by spaces\ in\ this\ filename to read

**cat spaces\ in\ this\ filename**

username : bandit3

Password: UmHadQclWmgdLOKQ3YNgjWxGoRMb5luK

**Bandit3 -> Bandit4**

Password is stored in a hidden file - inhere directory

[\*] Used cat cmd followed by .hidden to read

**cat .hidden**

username : bandit4

Password: pIwrPrtPN36QITSp3EQaw936yaFoFgAB

**Bandit4 -> Bandit5**

Password is stored in the human readable file- inhere directory

[\*] Used cat cmd followed by ./-file07 to read

**cat ./-file07**

username : bandit5

Password: koReBOKuIDDepwhWk7jZC0RTdopnAYKh

**Bandit5 -> Bandit6**

Password is stored under inhere directory- human readable,1033 bytes in size, not executable

[\*] Used cat cmd followed by  **./maybehere07/.file2** to read

**cat ./maybehere07/file2**

username : bandit6

Password: DxjZPULLxYr17uwoI01bNLQbtFemEgo7

**Bandit6 -> Bandit7**

Password is stored under the server- owned by user bandit7,owned by group bandit6,33 bytes in size

[\*] Used find cmd followed by / -user bandit7 -group bandit6 -size 33c to find and [\*] Used cat cmd followed by  **/var/lib/dpkg/info/bandit7.password** to read

**find / -user bandit7 -group bandit6 -size 33c**

**cat /var/lib/dpkg/info/bandit7.password**

username : bandit7

Password:

HKBPTKQnIay4Fw76bEy8PVxKEDQRKTzs

**Bandit7 -> Bandit8**

Password is stored in data.txt after ‘millionth’

[\*] Used cat cmd to read

**cat data.txt | grep "millionth"**

username : bandit8

Password: cvX2JJa4CFALtqS87jk27qwqGhBM9plV

**Bandit8 -> Bandit9**

Password is stored in data.txt

[\*] Used cat cmd to read

**cat data.txt | sort | uniq -u**

username : bandit9

Password: UsvVyFSfZZWbi6wgC7dAFyFuR6jQQUhR

**Bandit9 -> Bandit10**

Password is stored in data.txt in one of the human-readable strings, beginning with ‘=’ .

**strings data.txt | grep "="**

username : bandit10

Password: truKLdjsbJ5g7yyJ2X2R0o3a5HQJFuLk

**Bandit10 -> Bandit11**

Password is stored in data.txt containing base64 encoded data

[\*] Used cat cmd to read

**cat data.txt**

**cat data.txt | base64 --decode \\ to decode the password**

username : bandit11

Password: IfukwKGsFW8MOq3IRFqrxE1hxTNEbUPR

**Bandit11 -> Bandit12**

Password is stored in data.txt - all lowercase (a-z) and uppercase (A-Z) letters have been rotated by 13 positions

[\*] Used cat cmd to read

**cat data.txt | tr '[A-Za-z]' '[N-ZA-Mn-za-m]'**

username : bandit12

Password: 5Te8Y4drgCRfCx8ugdwuEX8KFC6k2EUu

**Bandit12 -> Bandit13**

Password is stored in data.txt -hexdump of a file thats repeatedly compressed

[\*]created a directory using /tmp- mkdir. For example: mkdir /tmp/prabhi123. After that copied the datafile using cp, and renamed it using mv .

**mkdir /tmp/prabhi123**

**cp data.txt /tmp/prabhi123**

**cd /tmp/prabhi123**

**file data.txt**

**xxd -r data.txt data1**

**file data1**

**mv data1 data2.gz**

**gzip -d data2.gz**

**file data2**

**mv data2 data3.bz2**

**bzip2 -d data3.bz2**

**file data3**

**mv data3 data4.gz**

**gzip -d data4.gz**

**file data4**

**tar -xvf data4**

**file data5.bin**

**tar -xvf data5.bin**

**file data6.bin**

**mv data6.bin data7.bz2**

**bzip2 -d data7.bz2**

**file data7**

**tar -xvf data7**

**file data8.bin**

**mv data8.bin data9.gz**

**gzip -d data9.gz**

**file data9**

**cat data9**

username : bandit13

Password: 8ZjyCRiBWFYkneahHwxCv3wb2a1ORpYL

**Bandit13-> Bandit14**

Password is stored in /etc/bandit\_pass/bandit14-read by user bandit14

[\*] using private SSH key

**ssh bandit14@localhost -i sshkey.private**

username : bandit14

Password: 4wcYUJFw0k0XLShlDzztnTBHiqxU3b3e

**Bandit14-> Bandit15**

We can get the password by connecting to port 30000 on localhost using the same password

**nc localhost 30000**

username : bandit15

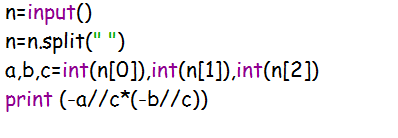
Password:BfMYroe26WYalil77FoDi9qh59eK5xNr

**Programming**

**1)1A- A. Theatre Square**

[\*] .py pgrm

Used split() and int() to convert the string -> integer to get the desired output.

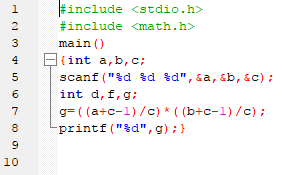


Full Code at:

<https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py>

[\*] .c pgrm

Used scanf() to get the input from the user.



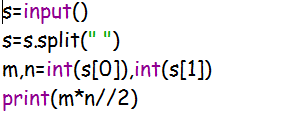
Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces\_C/1A.c](https://github.com/prabhigupta/bi0s/blob/master/codeforces_C/1028.c)

**2)50A - A. Domino piling**

[\*] .py pgrm

Used split() and int() to convert the string -> integer to get the desired output.

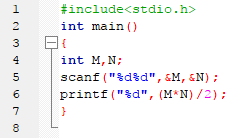


Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/50A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

[\*] .c pgrm

Used scanf() to get the input from the user



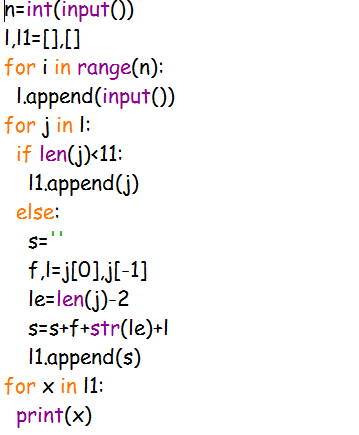
Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces\_C/50A.c](https://github.com/prabhigupta/bi0s/blob/master/codeforces_C/1028.c)

**3)71A- A. Way Too Long Words**

[\*] .py pgrm

Used len() to check the length of the input string and the concatenation operator, '+' to join multiple strings.

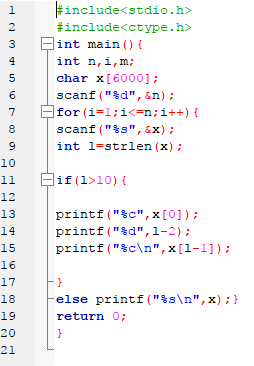


Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/71A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

[\*] .c pgrm

Used scanf() to get the input from the user and strlen() to get the length of the string.

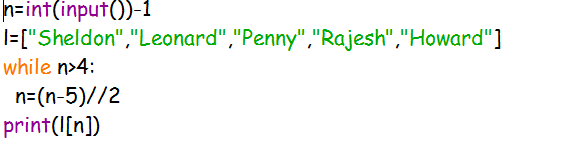


Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces\_C/71A.c](https://github.com/prabhigupta/bi0s/blob/master/codeforces_C/1028.c)

**4) 82A- A. Double Cola**

[\*] .py pgrm

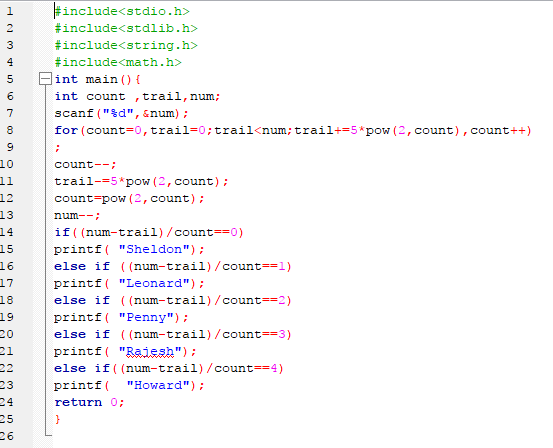


Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/82A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

[\*] .c pgrm

Used pow() to calculate the power raised to the base value.



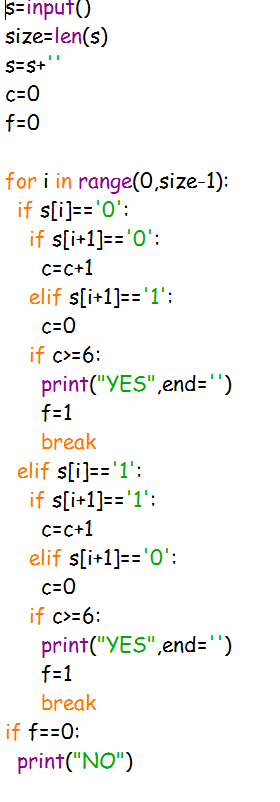
Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces\_C/82A.c](https://github.com/prabhigupta/bi0s/blob/master/codeforces_C/1028.c)

**5)96A- A. Football**

[\*].py pgrm

Used for loops to check whether the given case is dangerous or not.



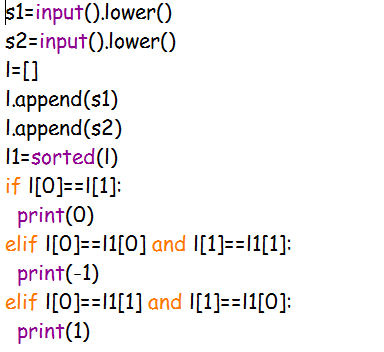
Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/96A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

**6)112A- A. Petya and Strings**

[\*].py pgrm

Used ‘==’operator to check the position of two strings .

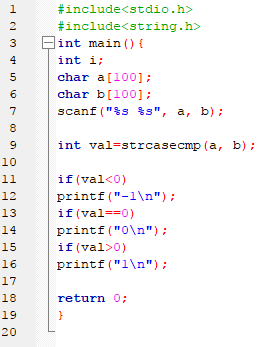


Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/112A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

[\*].c pgrm

Used to compares two strings irrespective of the case of characters.



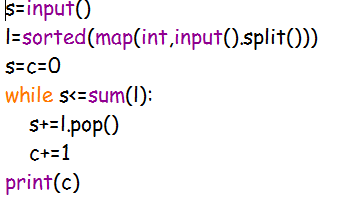
Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces\_C/112A.c](https://github.com/prabhigupta/bi0s/blob/master/codeforces_C/1028.c)

**7)160A - A. Twins**

[\*].py pgrm

Used map() to convert input string to int.

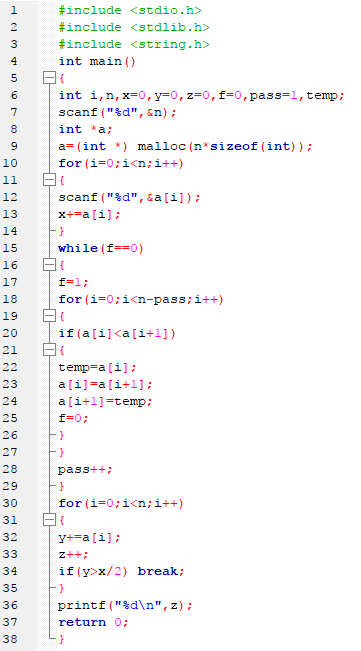


Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/160A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

[\*].c pgrm

Malloc() used to allocate a block of memory on the heap. sizeof() used to compute the size of its operand and It returns the size of a variable



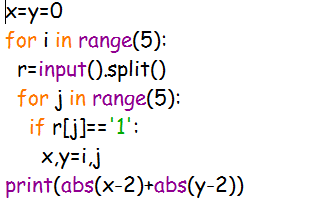
Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces\_C/160A.c](https://github.com/prabhigupta/bi0s/blob/master/codeforces_C/1028.c)

**8) 263A - A. Beautiful Matrix**

[\*].py pgrm

Used abs() to get the absolute value.

****

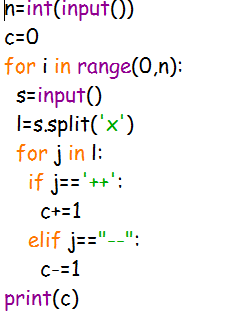
Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/263A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

**9)282A - A. Bit++**

[\*].py pgrm

Used split() to extract “++” & “--” symbol and for and if loop to calculate the desired output.



Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/282A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

**10) 339A - A. Helpful Maths**

[\*].py pgrm

Used sorted() to sort the list and join() to join multiple strings.

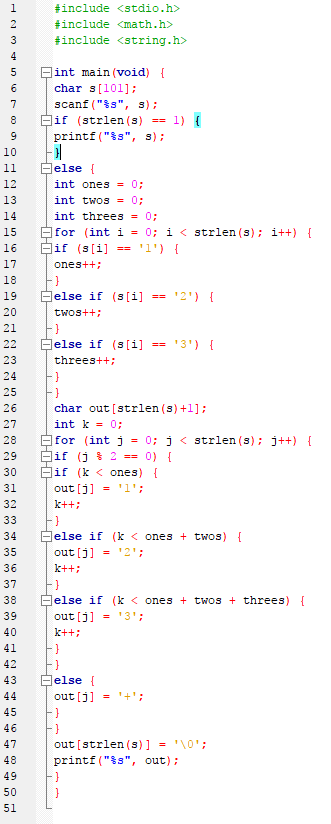


Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/339A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

[\*].c pgrm

Used strlen() to get the length of the string.



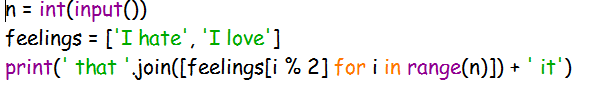
Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces\_C/339A.c](https://github.com/prabhigupta/bi0s/blob/master/codeforces_C/1028.c)

**11) 705A - A. Hulk**

[\*].py pgrm

Used join() and “+” to join multiple strings.

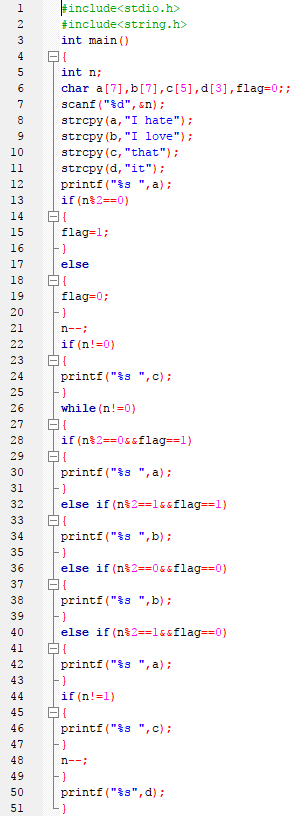


Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/705A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

[\*].c pgrm

used strcpy() to copy strings.



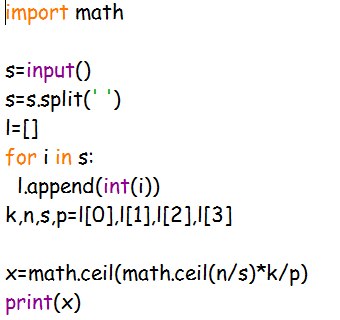
Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces\_C/705A.c](https://github.com/prabhigupta/bi0s/blob/master/codeforces_C/1028.c)

**12) 965 - A. Paper Airplanes**

[\*].py pgrm

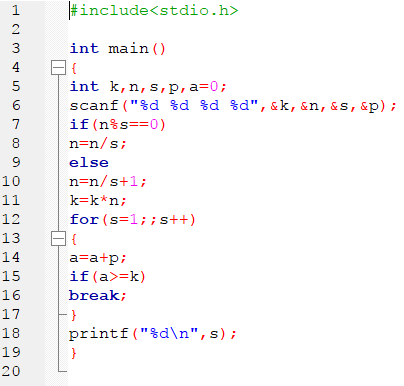
Imported math module for ceil() for getting the smallest integer not less than x



Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/965A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

[\*].c pgrm

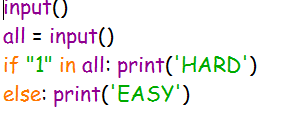


Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces\_C/965A.c](https://github.com/prabhigupta/bi0s/blob/master/codeforces_C/1028.c)

**13) 1030A - A. In Search of an Easy Problem**

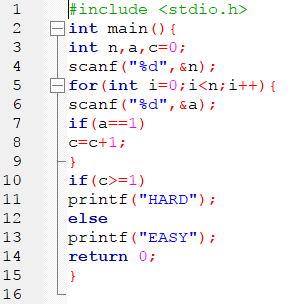
[\*].py pgrm



Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1030A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

[\*].c pgrm



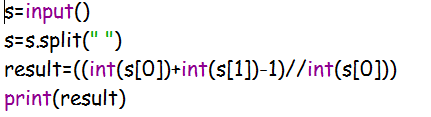
Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces\_C/1030A.c](https://github.com/prabhigupta/bi0s/blob/master/codeforces_C/1028.c)

**14) 1061A - A. Coins**

[\*].py pgrm

Used split() and int() to convert string input to integer



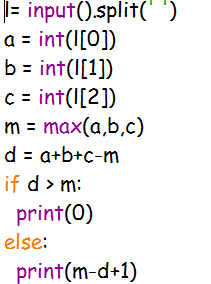
Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1061A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

**15) 1064A - A. Make a triangle!**

[\*].py pgrm

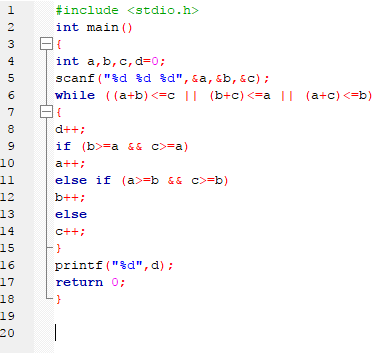
Used max() to get the longest length out of 3



Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1064A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

[\*].c pgrm

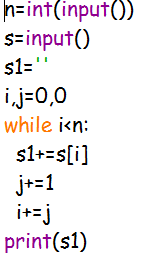


Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces\_C/1064A.c](https://github.com/prabhigupta/bi0s/blob/master/codeforces_C/1028.c)

**16) 1095A - A. Repeating Cipher**

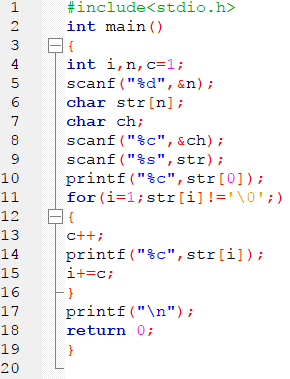
[\*].py pgrm



Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1095A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

[\*].c pgrm



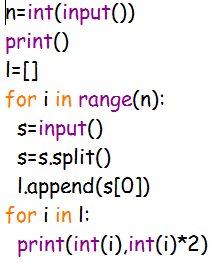
Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces\_C/1095A.c](https://github.com/prabhigupta/bi0s/blob/master/codeforces_C/1028.c)

**17) 1096A - A. Find Divisible**

[\*].py pgrm

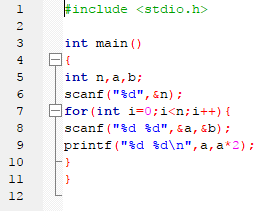
Used split() and int() to convert string input to integer



Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1096A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

[\*].c pgrm

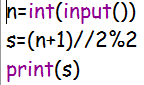


Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces\_C/1096A.c](https://github.com/prabhigupta/bi0s/blob/master/codeforces_C/1028.c)

**18) 1102A - A. Integer Sequence Dividing**

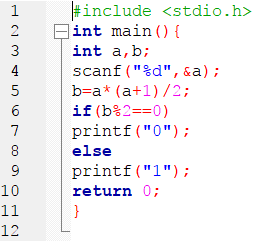
[\*].py pgrm



Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1102A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

[\*].c pgrm

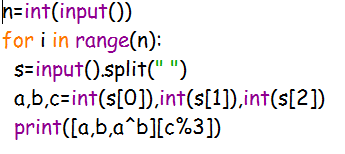


Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces\_C/1102A.c](https://github.com/prabhigupta/bi0s/blob/master/codeforces_C/1028.c)

**19) 1208A - A. XORinacci**

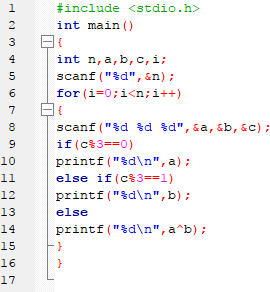
[\*].py pgrm



Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1208A.py](https://github.com/prabhigupta/bi0s/blob/master/codeforces%20task/1A.py)

[\*].c pgrm



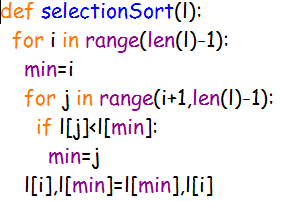
Full Code at:

[https://github.com/prabhigupta/bi0s/blob/master/codeforces\_C/1208A.c](https://github.com/prabhigupta/bi0s/blob/master/codeforces_C/1028.c)

**Algorithms**

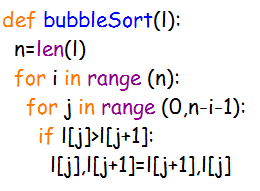
1. Selection Sort

Sorting algorithm which selects the least minimum element in the given array and displaced it with the ith element for every iteration



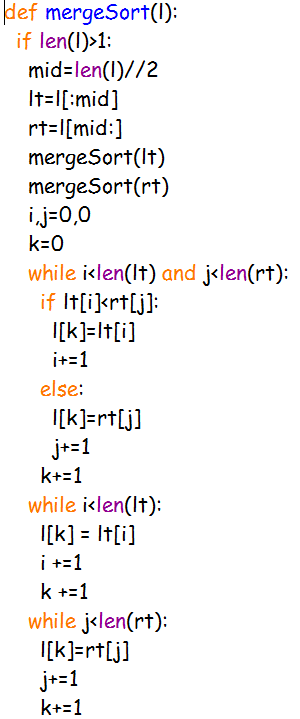
1. Bubble Sort

It is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order



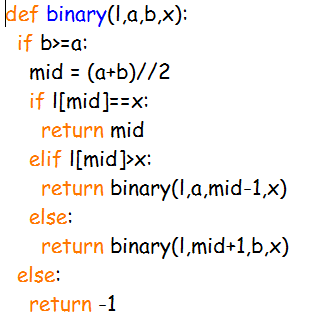
1. Merge Sort

It works by splitting the input list into two halves, repeating the process on those halves, and finally merging the two sorted halves together.



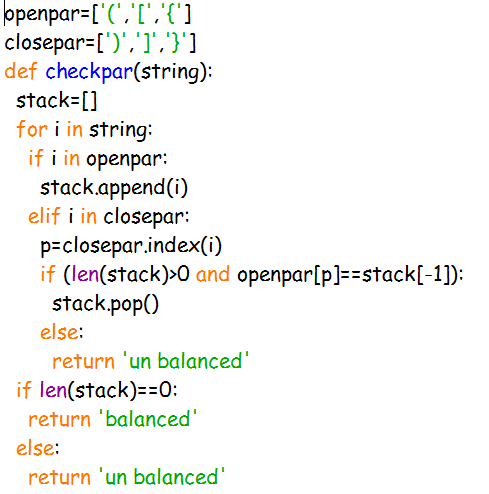
1. Binary Search

This search algorithm finds the position of a target value within a sorted array.



1. Check for Balanced Parentheses matching using Stack

This algorithm operated when an open parentheses is encountered push it in the stack, and when closed parenthesis is encountered



**Field Related Task**

**Implementation of shopping cart**

**PROJECT SYNOPSIS**

The Book store is a simple e-commerce application where users can select books, view the bookstore catalogue, and purchase books.

Two roles are defined for the system: customer and administrator. A customer would only be able to buy books or search for them. An administrator can add books to the inventory, query all the orders placed in the website and manage users.

Books are organized into various categories. All these categories are displayed in a user-friendly manner.

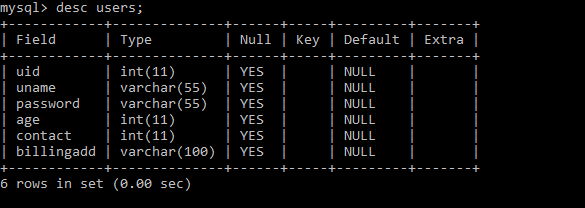
A guest can query for any book. To buy a book, the user must be logged-in. There is a ‘Buy book’ button under each book. This takes the user to the ‘Buy book’ page. If the number of copies available for the selected book is zero, appropriate message is displayed, otherwise, the customer information would be auto-populated in the Buy-Book page. The user can select the number of copies required. The total price is displayed to the user. By clicking on ‘Buy’, it displays success message.

A log of all the books sold including the details of the users who bought them are maintained. The admin would be able to query all the orders placed in the website.

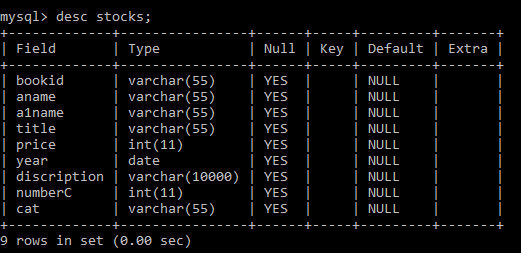
Python 3.7.4 has been used in the project. Tkinter module is used in the project for creating GUI (Graphical User Interface), which makes it easy to access and provides the click and view style. MySQL is used as backend database.

Table Structure of tables created and used for Book Store project are as below

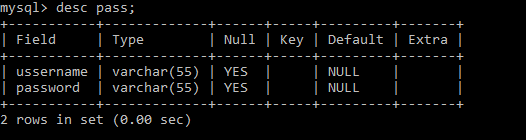
users:



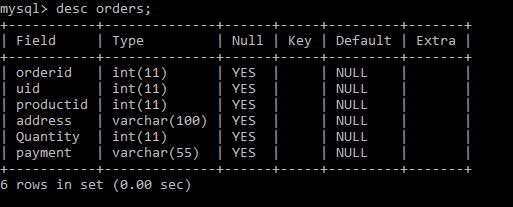
Stocks:



Pass:



Orders:



**OUTPUT**

Screenshot of various pages of Book Store are listed below

