# Word-Hunt

**CSA Group 03 Course Project**

|  |  |  |  |
| --- | --- | --- | --- |
| **Roll No.** | **Div** | **GR No** | **Name of Student** |
| 02 | A | 12010250 | Aahan Jain |
| 03 | A | 12010588 | Aarya Tiwari |
| 05 | A | 12011089 | Harshal Abak |
| 08 | A | 12011406 | Omkar Abhang |
| 09 | A | 12010599 | Abu Dujana Ansari |

## Introduction:-

This Project is a game of word-hunt where the code generates a random set of letters in a 14X30 grid and there are certain words hidden in that random grid that the user has to find.

# Algorithm:

First, we declare the data such as word pointers, the hidden words, etc. Then we set the index for word pointers and the direction of the words. We sort 30 columns and multiply by 2, the result is only even columns. To generate random words, we first sort a random number between 0 and DH, then do XOR with Al, then store that in dl register. Now using that data, we print the word stored in SI register. For D1, 0 = vertical, 1 = horizontal. If di is horizontal, print blank space. Then Get and set video mode, print content stored in AL. print content stored in Al, initiate screen, print character stored in DL.

data segment:

# initiate all data

seeds

words

word pointers

stack segment:

# initialize dw

code segment:

# set segment registers

main\_loop:

# call set\_word\_pointer

# call print\_board

#set index for word\_pointer

#word direction

fim:

#exit to operating system.

sort\_number:

# sort a number between 0 and DH

# XOR with AL

# store in dl

print\_word:

# print a word stored in SI

# DI: 0 = vertical, DI: 1 = horizontal

# if horizontal, print blank space

end\_check:

# check if pointer reached end

clear\_screen:

# get and set video code

set\_word\_pointer:

# set word pointers to different positions on screen

print\_custom\_character:

# print content stored in AL

# set colours and number of times to print

print\_board:

# get new letter

# check if it is letter

# print the letter

print\_character:

# print character stored in DL

random:

# store seconds in dh and cx as well

# ax = seed

# dx = 8405h

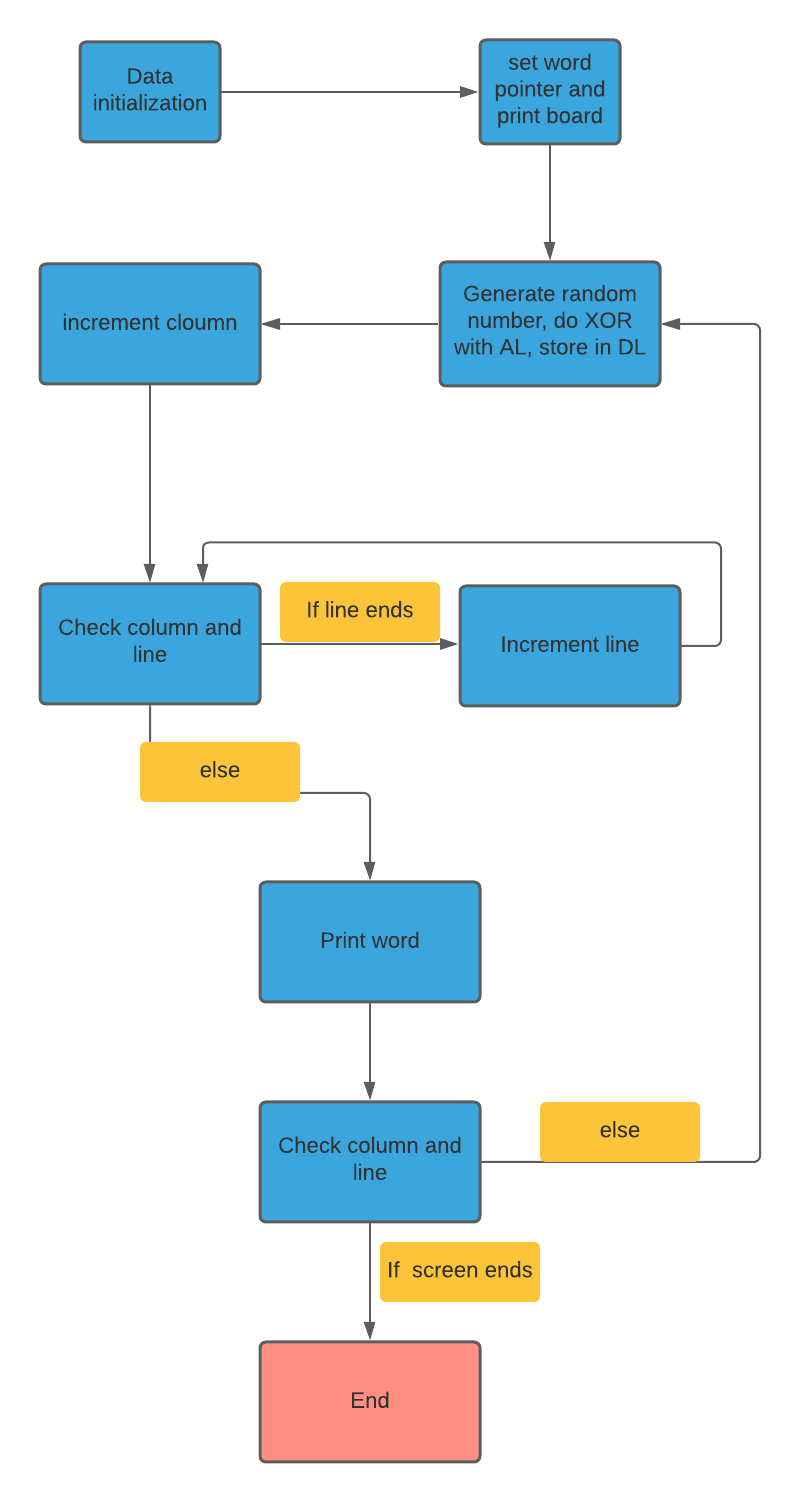
# multiply seed into dword dx:ax

# if new seed = old seed, alter seed

set\_cursor:

#set cursor on last word

## Flowchart



## Instructions Used:

mov – for copying the data between registers

call – for calling different functions

ends - for declaring the end of a particular function

add – for adding digits

mul – for multiplying digits

XOR – for doing binary XOR between digits

loop – for making code that is run repeatedly

cmp – to compare digits

push – to push elements in stack

pop – to pop elements in stack

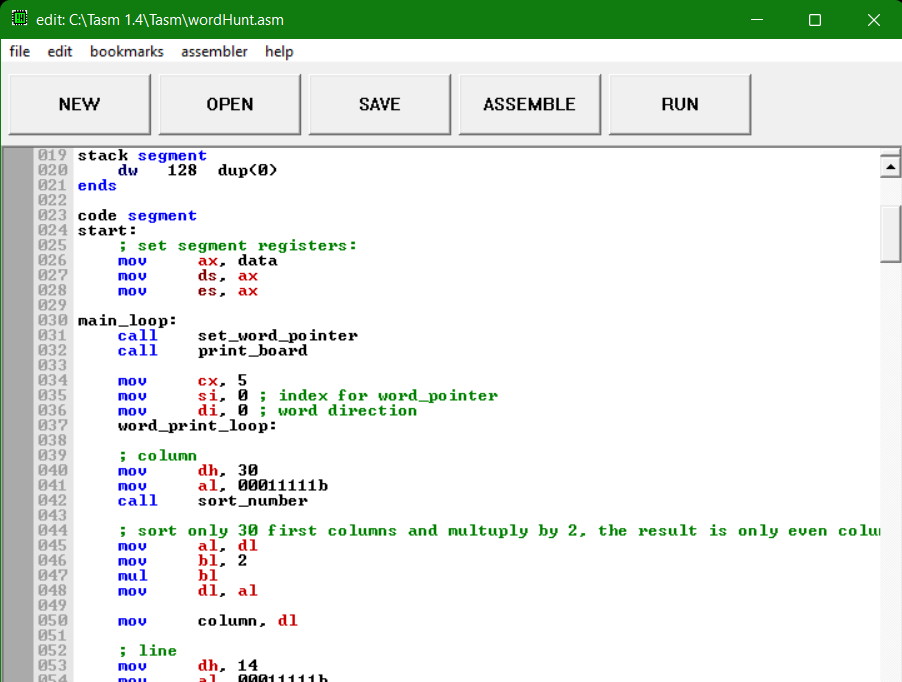
jmp – to jump to a certain line in code

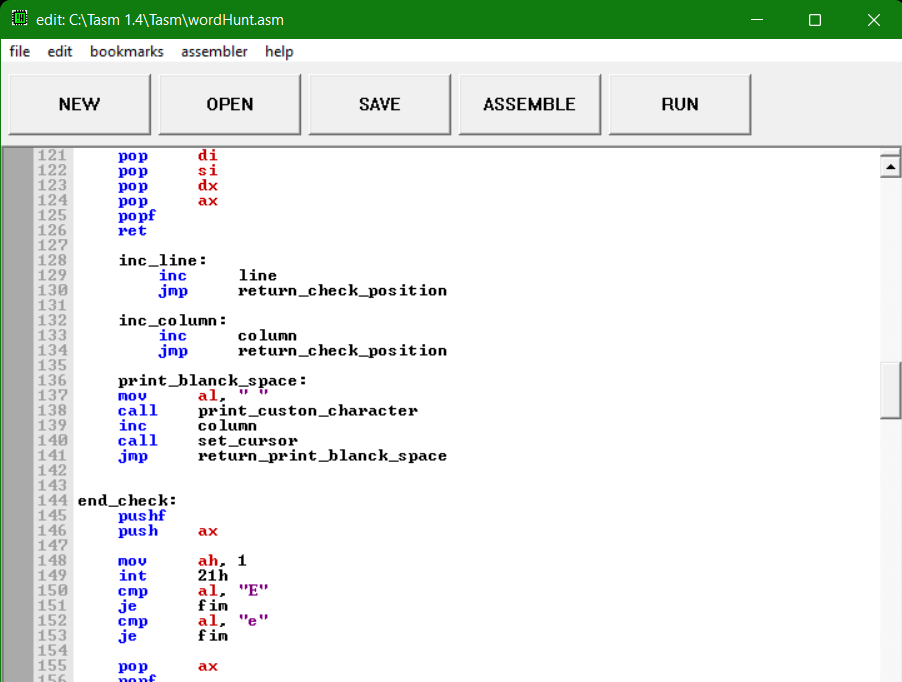
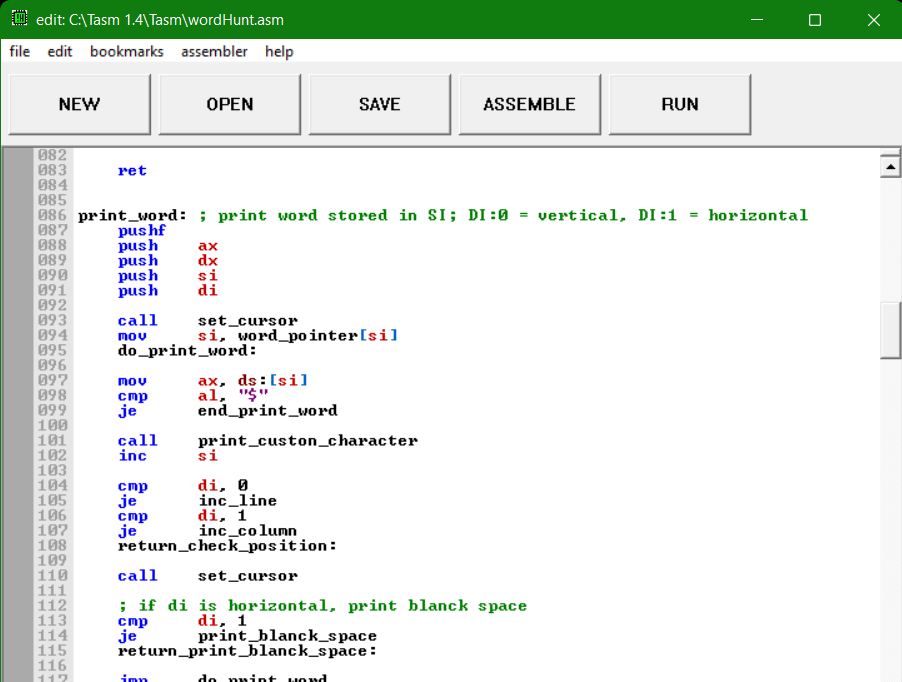
je – for doing if else statements

inc – for increment operation

ret – to return from a procedure

## Screen Shots

Code:- 



### Output:-

