VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belagavi, Karnataka 590018

17ISL67-FILE STRUCTURE MINI-PROJECT REPORT
on

FORTRESS OF GAMERS

Submitted in partial fulfillment of the requirement
for the award of the degree of

Bachelor of Engineering
in

Information Science & Engineering
by

BHARATH S (1BG17IS008)

B.N.M. Institute of Technology

Approved by AICTE, Affiliated to VTU, Accredited as grade A Institution by NAAC.
All UG branches – CSE, ECE, EEE, ISE & Mech.E accredited by NBA for academic years 2018-19 to 2020-21 & valid upto 30.06.2021
Post box no. 7087, 27th cross, 12th Main, Banashankari 2nd Stage, Bengaluru- 560070, INDIA
Ph: 91-80- 26711780/81/82 Email: principal@bnmit.in, www. bnmit.org
Department of Information Science and Engineering

2019 – 2020

Fortress of Gamers

Introduction

Fortress of gamers is a web page that provides all required information of all the games that are released. The website provides information of what genre the game belongs to, who all the characters are in the game, the voice actors and the company that created the game.

The web page is made for the sole purpose of helping gamers get information about the games the love so as to understand the game better so as to improve their experience.

The application is a web application where the user interacts with the user interface where once the user enters the name of the game, information of that game is displayed by extracting the information from the respective files.

File Structure

File structure is an organization of data in Secondary Storage Device in such a way that it minimizes the access time and the storage space. It is a combination of representation for data in files and of operations for accessing the data. Early in the computing history, secondary storage was in the form of magnetic tape and punched cards. Storage was cheap but access was limited to sequential. In 1956, IBM introduced the RAMAC magnetic disk device. Data could be accessed directly instead of sequentially. This was the drawn of the study of file structures. Advances in OS gave rise to more research on operating systems. The next analogy that was come up was the Direct Access which is the analogy to access to position in array. Indexes were invented and list of keys and pointers were stored in small files. This allowed direct access to a large primary file. But as the file grows the same problem arise as with the primary memory. Tree structures emerged for main memory in 1960's. This involved balanced and self-adjusting Binary Search trees (BST) eg: AVL trees (1963) . In 1979, a tree structure suitable for files was invented: B trees and B+ trees good for accessing millions of records with three or four dist. accesses. Theory on Hashing tables were developed over 60's and 70's good for those files that do not change much over time. Expandable and dynamic Hashing were invented in late 70's and 80's which provided for one or two disk accesses even if file grow dramatically.

A good File Structure aims at:

- Fast access to greater capacity.
- Reduce the number of disk accesses by collecting data into buffers, blocks or buckets.
- Manage growth by splitting these collections.

The application contains 4 file Games, Publisher.
The file is structured in the way that when a game is searched all the related information required can be fetched from the other files with high efficiency.

The storing of data and retrieving the data from the files are done using hashing method.

SYSTEM REQUIREMENTS

To be used efficiently, all computer software needs certain hardware components or other software resources to be present on a computer. These prerequisites are known as (computer) **system requirements** and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended. With increasing demand for higher processing power and resources in newer versions of software, system requirements tend to increase over time. Industry analysts suggest that this trend plays a bigger part in driving upgrades to existing computer systems than technological advancements. A second meaning of the term of System requirements, is a generalization of this first definition, giving the requirements to be met in the design of a system or sub-system.

Hardware Requirements:

- Hardware: DualCore
- Hard Disk: 50 GB
- Speed: 1.4 GHz
- RAM :1GB
- Key Board: Standard Keyboard
- Touch Pad: Button Mouse
- Monitor: LED

Software Requirements:

- Operating System: Windows
- IDE: PyCharm Community Edition 2019.3.3 x64
- Technology: Python
- Web Technologies: Html, CSS

Technologies used in project:

- PyCharm IDE as front end
- Files as back end

File organization

- The application has 4 files Games, Publisher.
- The data are stored using hashing method.
- Every file has key word that relates that file to other files in the application.
- There are 4 methods of adding structure to files, which are:
  - Force the fields into a predictable length.
  - Begin each field with a length indicator.
  - Place a delimiter at the end of each field to separate it from the next field.
  - Use a "keyword = value" expression to identify each field and its contents.

  The method used in this application is place a delimiter at the end of each field to separate it from the next field.
- The data stored are all of variable length.

Methodology

0(1) access to files means that no matter how big the file grows, access to a record always takes the same, small number of seeks. By contrast, sequential searching gives us O(N) access, wherein the number of seeks grows in proportion to the size of the file.

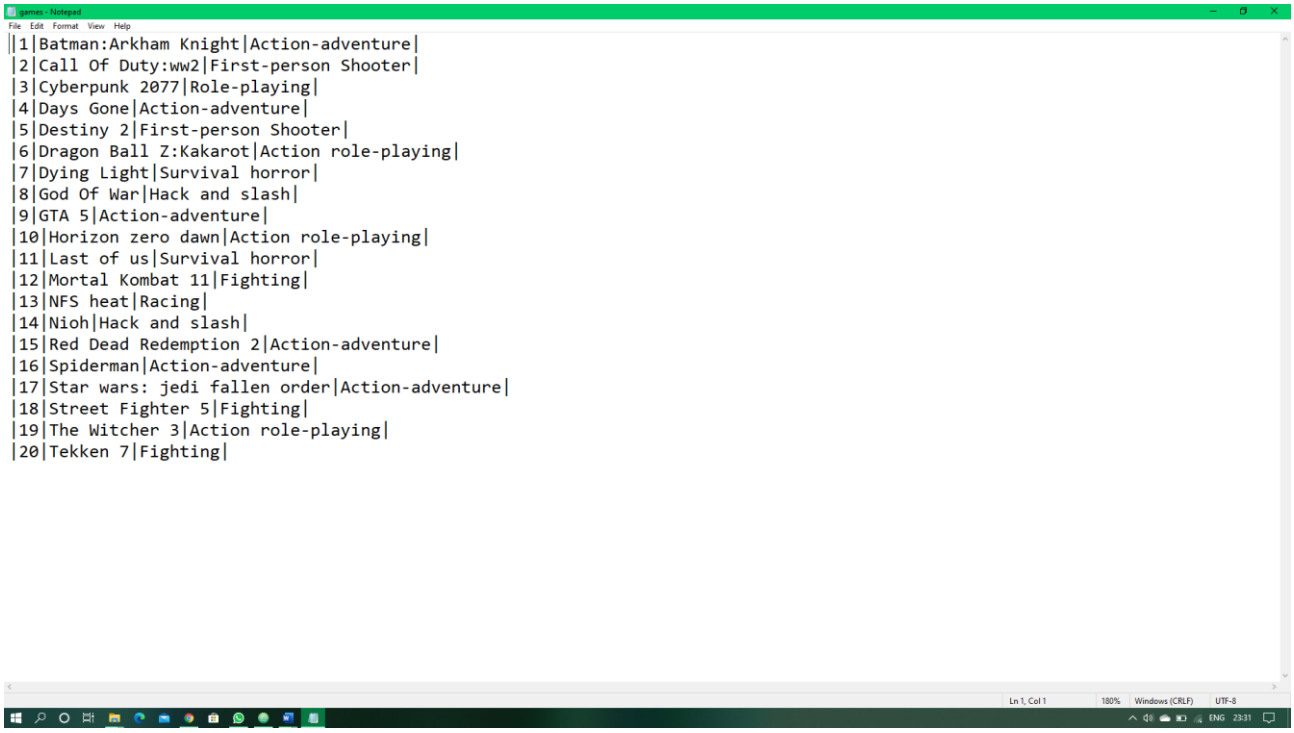
Hence the method used in the application is Hashing. A hash function is like a black box that produces an address every time you drop in a key. More formally, it is a function h(K) that transforms a key K into an address. The resulting address is used as the basis for storing and retrieving records. For example, the key LOWELL is transformed by the hash function to the address 4. That is, h(LOWELL) = 4. Address 4 is said to be the home address of LOWELL. Hashing is like indexing in that it involves associating a key with a relative record address. Hashing differs from indexing in two important ways:

- With hashing, the addresses generated appear to be random-there is no immediately obvious connection between the key and the location of the corresponding record, even though the key is used to determine the location of the record. For this reason, hashing is sometimes referred to as randomizing.
- With hashing, two different keys may be transformed to the same address so two records may be sent to the same place in the file. When this occurs, it is called a collision and some means must be found to deal with it.

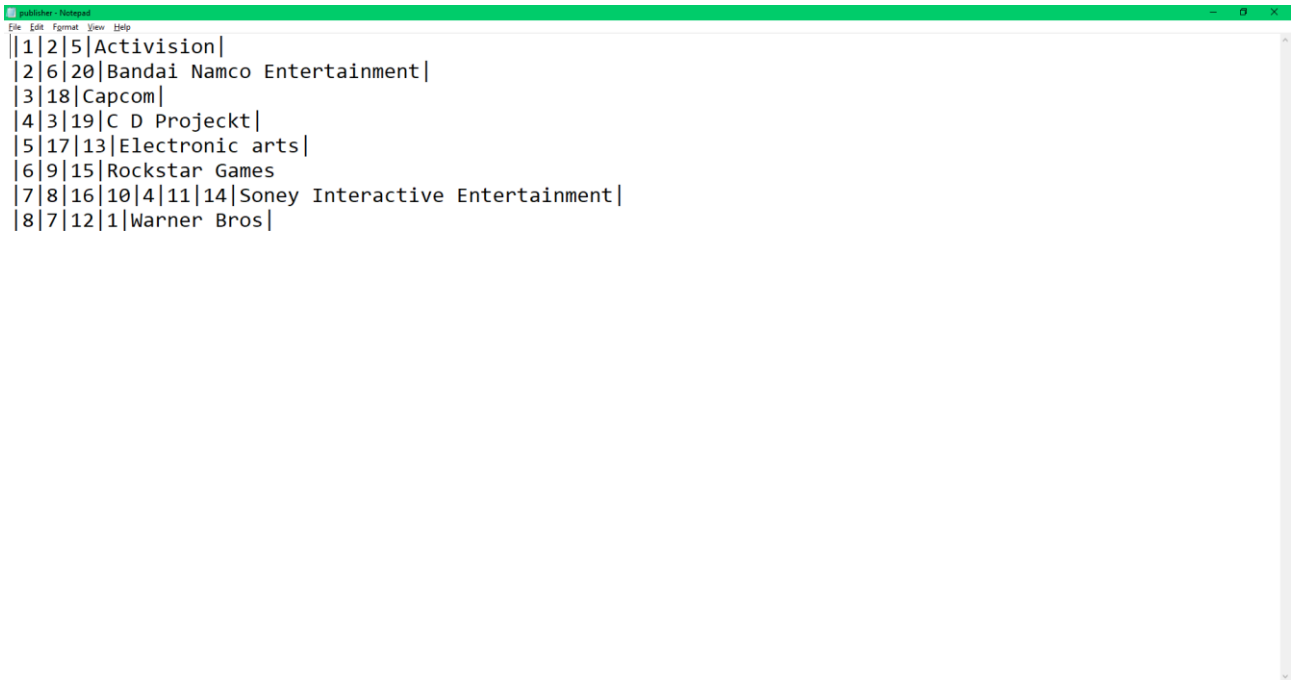
Snapshots of Files

This session consists of the screenshots about the game's records, that describes files used in the Application.

File 1: Games



File 2: Publisher



Snapshots of Application

This session consists of the screenshots of the basic GUI of Fortress of gamers application, that helps us perform certain specific actions.