ICT1007 Assignment (AY2018/2019)

Due: 1/4/2019 23:59

This assignment is to simulate the disk allocation and the free-space management implementation of a file system. The objective is to evaluate the various methods to determine its efficiency (i.e. speed in accessing file content) and flexibility (i.e. ability in keeping different file sizes). You are to form a team of 5/6 students and implement the following with a program. The program must be written in C.

Below are what you need to implement:

- Volume control block: this block will reflects the total number of blocks, number of free blocks, block size, free block pointers or bit map array of free block
- Directory structure: You needs a directory structure to keep track of the basic file information and the location of where the file content is stored. You can either use the linear or the hash table method.
- Disk blocks allocation: The program needs to implement all three standard disk blocks allocation methods plus a new method design by your team (i.e. Contiguous, Linked, Indexed and new method).
- Free-space management: You can use either bit-map or linked list methods to determine the free blocks locations.

The physical store is represented by an array of 128 entries with <user input> entries as a disk block. Table 1, as shown below, is a sample instance of the first 6 blocks of the physical store containing two different files. Note that this sample is using 5 entries per block. The directory structure is stored in the first block showing the information of the 2 files. In this example the first number, 100, is the file identifier. The second number is the starting index of the first data block (i.e. 1) and the second number is the index of the ending data block which is 4.

Index	Block	File Data
0	0	<v.ctrl B>
1	0	100,1,4
2	0	200,5,5
3	0	
4	0	
5	1	101
6	1	102
7	1	103
8	1	104
9	1	4
10	2	
11	2	
12	2	
13	2	
14	2	

Index	Block	File Data
15	3	
16	3	
17	3	
18	3	
19	3	
20	4	105
21	4	106
22	4	
23	4	
24	4	
25	5	201
26	5	202
27	5	203
28	5	
29	5	

Table 1

File is represented as a series of integer number ranging from 100 to 9999. The first number is taken as the file identifier (i.e. 100, 200, … 1000, 1100, …., 9800, 9900). The content of the

file is represented as successive integer number increment from the file identifier (i.e. 1101, 1102, 1103, …).

Each entry in the array holds one number. The number can be either the file content or a pointer representing the block number of another block. This will apply to all the array entries except the directory structure which can contain multiple attributes as shown in the above example. This structure will vary according to the allocation method. But each entry in the directory structure can only contain the information for one file.

Below are the key steps of what the program should do:

1. Your program will start by simulating the formatting process by asking the user for the required block size or number of block. The entire disk space needs to be formatted accordingly.

2. Your program will read a csv file as the input in the following format. The first field is the file operation which includes add, read or delete a file. The second filed is the file identifier and the remaining are the data of the file. Each line represents an operation the program needs to perform to a single file.

   add, 100, 101, 102, 103, 104, 105, 106
   add, 200, 201, 202, 203
   read, 106
   delete, 200

3. Based on a disk block allocation method, the program will complete the operation as stated and output the status of the action. The add operation will result in two output messages. One is a message for locating free block. Besides the listing of the free blocks, it also shows the "time" required to locate the free block. "Time" is defined as the number of times the program needs to traverse the array entry. The second message is to show the actual file location with the block identifiers of all the allocated blocks. The list should be in a proper sequence of how the file is stored. Output an error if the file cannot be stored because there is not enough space.

   The number after the read operation indicates the file content (i.e. 106) that need to be read. Output which blocks and addresses are being read and also the access "time". For the delete operation, just indicates that the file is deleted and reports which blocks are freed. Output appropriate error message if the file cannot be added (file exist) read (no such file) or delete (no such file). Below is an example of the output:

   Adding file100 and found free B1,B4
   Added file100 at B1(101,102,103,104), B04(105,106)

   Adding file200 and found free B5
   Added file200 at B5(201,202,203)

   Read file100(106) from <you will decide how it can be processed>

   Deleted file200 and freed B5

4. The final output from the program should be a disk map showing the full contents of the physical store (i.e. all 128 entries). You can use table 1 as a guide. It should also show the total number of added files, the overall size (i.e. number of integer) and the "time" it takes in the addition. This should also apply to the read operation.

5. Pick another disk allocation method and repeat the process again from step 2 or 3. The program should simulate using all 4 methods.

Below are what you need to include in a report:

The report must start with a title page showing the ID and name of all the team members. It should also contain the contribution percentage and a brief description of the tasks of each member. Make sure all the team members sign the submitted hardcopy.

Team ID xx				
ID	Name	Percentage	Task Description	Signature

After the title page includes a self-diagnosis and evaluation statement giving the following details for each requirement: what have been fully working, and what have been completed but not fully working, and what are not been fully completed or not attempted. This statement is essential.

The main section of the report is to show the overall program design, the program structure of the respective components (i.e. directory structure, disk block allocation and free space management). Explain clearly how each of these components work and how they have assisted other components to fulfil the objective. Highlight any special construct or design you have implemented to make the program more efficient and flexible.

The next section should contain the screenshots of all the outputs to demonstrate your program is working as required. Besides the given sample data, you should also design other test data to ensure that your program is robust and works in any condition. Document all the testing in this section.

The final section is to analyse your finding. Tabulate the results that you have obtained. Evaluate the performance of each disk allocation methods. Discuss in detail the advantages and the disadvantages of each method. Use the result you have obtained to support your discussion.

Append the source code of your program as an appendix in this report. The source code must be properly indented, commented, and formatted.

ICT1007 Assignment (AY2018/2019)

Due: 1/4/2019 23:59

Below are what you need to submit:

Your assignment must be submitted in the form of a single zip file via moodle on or before 1st April 2019 (one submission per team). The zip file must be named as "ICT1007-YourTeamID(IDs of all students).zip". You will also need to submit a hardcopy before your demonstration.

The zip file contains the following:

- Report in MS Words format and named the file as "ICT1007-YourTeamID(IDs of all students).docx".
- Your program
- All the test files

Evaluation session:

An evaluation session will be arranged after your submission for your team to demonstrate your work. The instructor will use a different test file with the same format during the evaluation. Your program must work for the evaluation.

Marking Guide:

Demonstration (with tutor test set)		
Formatting	5	
Directory structure	5	
Allocation: contiguous	5	
Allocation: linked	5	
Allocation: indexed	5	
Allocation: New method	10	It must be unique where no other team has implemented
Free-space management	5	
Integration of all the components	5	
Presentation of the output	5	
Testing	5	
Code style	5	
Report and analysis	40	Clearly justify all design considerations.