Name – **Malav Doshi , Herik Patel**

Systems programming

Professor – Dr Francisco

Assignment 1

Description :

In this assignment we used file system API. The Program recursively goes through the files in the directory and takes data stored in the files and converts the data in Huffman code. Min Heap is used to store the data and the data is stored in in leaf node. The data given to us is compressed and stored in file with extension of .hcz.We take the compressed code and decompress the code and store it inside the file. The list of function that were used and their purpose is mentioned below:

1. **void addNode(struct node* nd)**- The function adds node to the linked list. Function returns nothing and takes a node as argument and adds it in a linked list.

2**. void addNodeCpy(struct node* nd)**-The function adds node to the linked list. Function returns nothing and takes a node as argument and adds it in a linked list.

3. **struct heapNode* newHeapNode(char* str, int size, int freq)**-It builds new heap node for the heap and takes char* ,two integer arguments and returns a heap Node pointer.

4. **struct node* newNode(int size)**- The function builds a new node on basis of size needed to store in the node and takes integer as argument. A String is stored in the node. Function returns a node pointer.

5. **void findRepeat()**-The function is used to figure out if a token repeats in the list or not.

6. **void insertCopyList()**- The function checks if values are repeating and if it finds any repeating values it increments heap array size. The function returns nothing and no argument is passed.

7. **void swapNodes(struct heapNode* node1, struct heapNode* node2)** - The function swap values in two given heap nodes and takes two argument both of heap node type and does not return anything.

8**. int getLeftChild(int index)** -The function is used to get left child in the heap, takes integer as argument and returns integer value if child is not found it returns -1;

9. **int getRightChild(int index)** -The function is used to get left child in the heap, takes integer as argument and returns integer value if child is not found it returns -1;

10. **int getParent(int index)** -The function is used to get parents of child node. The function takes an integer as argument and returns an integer.

11. **void heapify(struct heapNode arr[], int index)** - This is used during building min heap to sift up a new node that is added to the heap. Return type is void.

12. **struct heapNode* getMinimum(struct heapNode arr[])** - This is used to get the minimum element of the min heap. It takes array as argument and return heap node pointer.

13. **struct heapNode* buildHuffmanCode(struct heapNode arr[])** - The function builds Huffman code using heap and returns a heap node pointer and takes array as an argument.

14. **void storeToList(int arr[],char* token, int size)** - The function stores Huffman code to the list and the argument passed is of integer array char pointer and an integer and function does not return anything.

15. **void getHuffmanCode(struct heapNode* root, int arr[], int top)** - The function gets Huffman code from the heap. The argument passed are heapNode pointer , an integer array and an integer.

16. **void tokenizer(char* string)** -This function takes char pointer as input and which will be a string and it will tokenize the string on basis of given delimiters and stores the string in linked list. Function does not return anything.

17. **void freeList()**- The function deallocates memory which has been allocated to the linked list. Function does not take any argument and does not return any value.

18. **void writeToFile(struct node* hd, char* fileName)** - The function is used to open files and write down Huffman code in ".hcz" file. The argument passed is node pointer and char pointer which is filename. The function does not return anything.

19. **void writeToCodeBook(struct node* hd)** - The function is used to write Huffman code to Huffman codebook. The argument passed is node pointer. The function does not return anything.

20. **void compressFile(char* path, char flag)** - The function compresses the string present in the file using tree structures and does not return anything. The argument is of character pointer and a character.

21. **int searchList1(char* code)** – The function checks if the code from the ".hcz" is in the list or not. Returns 1 if the code is present in the list else returns 0.

22. **void writeToOriginalFile1(char* file, char* code)** - The function writes decompress code to the original file. Two character pointer are passed in the argument and the function has no return value.

23. **void decompressFile1(char* path, char* codeBook)** - The file which was compressed using Huffman codebook is decompressed using this function the function has no return value and arguments passed is of two character pointers.

24.**void addHeapNode(struct heapNode arr[], struct heapNode* node)** – adds the heapNode* node to the given heap. Return type void.

25. **void findFrequency(struct node* nd) –** The function is used to figure out how many times a token repeats in a list. Return value is void.

26. **char* putHCZ(char* file, int size) -** Adds .hcz extension to the given file. Returns a string which has a file extension of .hcz .

27. **void editCodeBook(char* codeBook) –** Edits the HuffmanCodebook after it is used in recursive decompression. Return type void.

28. **void printFiles(char* basePath, char flag) –** The function is called for recursive compression of files or recursive building of Huffman Codebook.

29. **void recursiveDecompress(char* basePath, char* codeBook) –** The function is used for recursive decompression of files. Return type is void.