|  |  |
| --- | --- |
| **Name** | **Hatim Yusuf Sawai** |
| **UID no.** | **2021300108** |
| **Experiment No.** | 10 |

|  |  |
| --- | --- |
| **AIM:** | Implement various operations on files to solve a given problem. |
| **Program 1** | |
| **PROBLEM STATEMENT:** | A publishing company holds in a file details of all the books they publish. However, in the future, they wish to maintain two distinct files (i) paperbacks (ii) hardbacks. Write a program which reads a file containing details of both paperback and hardback books and creates two files as specified above. Assume that the first character in each input record indicates if the book is paperback(p) or hardback(h) or both(b). |
| **ALGORITHM:** | 1. START  2. Define structure book\_t containing character variable type,  character arrays name, and author  3. Define int function print having character array filename[] as a  parameter.  4. Initialize three file pointers fp,fp1, and fp2.  5. Use fp to open filename in reading mode  6. If fp is equal to NULL  print file not found  return 0  7. Initialize I to 0  8. Define book\_t variable b  9. Use fp1 to open Paperbacks.txt in write mode and fp2 to open  Handbacks.txt in write mode  10. While(file scan of fp for b.type, b.name, and b.author is not  equal to EOF)  If(file scan is equal to ‘p’)  file write of fp1 printing b.name and b.author  else If(file scan is equal to ‘h’)  file write of fp2 printing b.name and b.author  Else If(file scan is equal to ‘b’)  file write of fp1 printing b.name and b.author  file write of fp2 printing b.name and b.author  11. Close fp1  12. Close fp2  13. Close fp  14. Return 0  15. Define int main()  16. Declare character array filename[20]  17. Input filename  18. Concatenate filename with “.txt”  19. I = print(filename)  20. Return 0  21. STOP |
| **PROGRAM:** | #include<stdio.h>  #include<string.h>  typedef struct book  {      char type;      char name[50];      char author[50];  }book;  int sort(char file*[]*)  {      FILE \*fp,\*fp1,\*fp2;      fp = fopen(file,"r");      if(fp==NULL)      {          printf("File not found!\nMake sure you entered the correct file name.");          return 0;      }      book b;      fp1 = fopen("Paperbacks.txt","w");      fp2 = fopen("Handbacks.txt","w");      while(fscanf(fp," %c,%[^,],%[^\n]\n",&b.type,b.name,b.author)!=EOF)      {          if(b.type=='b')          {              fprintf(fp1,"%s, %s\n",b.name,b.author);              fprintf(fp2,"%s, %s\n",b.name,b.author);          }          else if(b.type=='p')              fprintf(fp1,"%s, %s\n",b.name,b.author);          else if(b.type=='h')              fprintf(fp2,"%s, %s\n",b.name,b.author);      }      fclose(fp);      fclose(fp1);      fclose(fp2);      printf("Files have been sorted successfully!");      return 0;  }  int main()  {      int i;      char file[20];      printf("Enter name of file to be sorted: ");      scanf("%s",file);      strcat(file,".txt");      sort(file);      return 0;  } |
| **RESULT:**  **File Containing Records:**  **Paperbacks file:**  **Hardbacks File:**  **Output:** | |
| **Program 2** | |
| **PROBLEM STATEMENT:** | Set up a file containing vehicle records which hold registration number and owner information (name and address). Write a program which, given a vehicle’s registration number, will rapidly retrieve and print the owner information. |
| **ALGORITHM:** | 1. START  2. Define structure vehicle\_v containing character arrays reg[7],  name[30] and addr[30] as variables  3. Define int function print with character array filename[] as a  parameter.  4. Define FILE pointer fp  5. Use fp to open filename in reading mode  6. If fp is equal to NULL  print File not Found  return 0  7. Close fp  8. Declare vehicle\_t variable v, integer variables n and flag =1,  and character array rg[7]  9. Input no of searches n  10. Loop from I = 0 to 6  a. Use fp to open filename in reading mode  b. Input registration number rg  c. While (file scan of fp for v.reg, v.name and v.addr) is not  equal to EOF  If (strcmp(v.reg,rg) is equal to 0)  print v.reg, v.name, and v.addr  flag = 0  break  d. If (flag)  print not found  e. Close fp  11. Return 0  12. Define int main()  13. Input character array filename  14. Concatenate “.txt” to filename  15. Int I = print(filename)  16. Return 0  17. STOP |
| **PROGRAM:** | #include<stdio.h>  #include<string.h>  typedef struct vehicle  {      char num[8];      char name[20];      char address[20];  }vehicle;  int retrieve(char file*[]*, char reg*[]*)  {      int flag=0;      FILE \*fp;      fp = fopen(file,"r");      if(fp==NULL)      {          printf("File not found!\nMake sure you entered the correct file name.\n");          return 0;      }      vehicle v;      while(fscanf(fp," %[^,],%[^,],%[^\n]\n",v.num,v.name,v.address)!=EOF)      {          if(strcmp(v.num,reg)==0)          {              printf("Name: %s\nAddress: %s\n",v.name,v.address);              flag=1;              break;          }      }      if(flag==0)          printf("Record Not Found!\n");      fclose(fp);  }  int main()  {      char file[20],reg[8];      int flag=1;      printf("Enter name of file to be searched: ");      scanf("%s",file);      strcat(file, ".txt");      while(1)      {          printf("Enter the registration number: ");          scanf(" %s",reg);          retrieve(file,reg);          printf("Do you want to search another record?(yes=1/no=0): ");          scanf("%d",&flag);          if(flag==0)              break;      }      return 0;  } |
| **RESULT:**  **Vehicle-Records File:**  **Output:** | |
| **Program 3** | |
| **PROBLEM STATEMENT:** | C program to replace the specified line in an existing text file. Read the line number from the user, and then replace the content-specific line with new text in the existing file. Then print the modified content file. |
| **ALGORITHM:** |  |
| **PROGRAM:** |  |
| **RESULT:**  **Original File:**  **Modified File:** | |
| **CONCLUSION:** | In this experiment, we learnt how to handle files in C. We learnt how to open a text file, read it, write it/overwrite it and search and replace text in the file. |