Project Report

On

DEVELOPMENT OF MERRIT LIST SOFTWARE IN C

By

ABHIJEET SANJIV BONDE

(E. No. 19103002)

Jaypee Institute of Information Technology, Noida

ABSTRACT:

On this planet, every academic institute needs a robust software which, based on some variable parameters can display the merit list based on marks the students have obtained. So, In this project an application is developed using C language which reads the source information from a text file and calculates the total marks and their percentage and then based on the Users choice, prepare the merit list and displays the same as well as can store the merit list into a text file.
In this project I have used the concepts of C language which majorly deal with Linked List, Strings, Numerical data and Sorting process. File I/O are also used in certain operation in order to get the best results and for the program to be user friendly.

Existing System:

Many more existing systems use the manual and are hand run processes to prepare the merit list and declare the result. The User has to manually enter the Name, Enrollment number, and Marks in each subject, then the software is ready to calculate the percentage and merit list according to the marks. This process is very slow to give the result. Such kind of software's are easily prone to making errors. When a parameter has to be added to such software, it has to be redesigned, adding more efforts. The more the students, the more tedious the work is.

Proposed System:

To overcome the Problems that the Existing systems face, this program is designed to take the input from a user defined text file and, the automated generated merit list is stored in a text file which makes the application versatile to use for the user. This makes the user easy to use the application. The program also keeps the record of number of students. Based on the data that has been entered in the text file, the application sorts the data and makes a Merit List. Since the User has to enter the data only once, it makes the application less prone to errors. When the Merit List is prepared, the User can Display

it or store it in a File. When a parameter or a record has to be added to the system, minimal changes are required. Number of students in not a issue for this system. The system can be easily scaled according to the needs.

Advantages:

1. Very fast, accurate, and robust.
2. No need of any extra manual effort.
3. With just minimal information user can operate the application.
4. Easy to operate
5. Options available for sorting the data on different parameters.
6. A User can create the merit list for all subjects separately, once the marks are entered in the file.

REQUIREMENTS:

1. The User has to input the data into a text file.
2. Data consists of Serial Number, Name of the student (First name, Middle name, and the Last name), Marks of 5 subjects (marks 1, marks 2, marks 3, marks 4, marks 5) with spaces in between each field.
3. Run the executable file.
4. Choose an option from the menu displayed.
5. If the user wants to store the Merit list, an option is available.
6. After the User has finished, Memory allocated to the linked list has to be freed to avoid any future errors.

MODULE DESCRIPTION:

As, with a well-organized display and system, it is better to navigate through the application. So, the application has been broken down in some simple functions.

1. MENU
2. CREATE LINKED LIST
3. SORTING ON VARIOUS PARAMETERS
4. DISPLAY
5. CREATE OUTPUT FILE
6. FREE MEMORY AND EXIT

MENU:

There are a lot of applications out there with just one UI. In this application also there is just one User Interface which is known as menu. It is totally based on C language. It has the following options.

```
        MENU
1 TO SORT ON MARKS 1
2 TO SORT ON MARKS 2
3 TO SORT ON MARKS 3
4 TO SORT ON MARKS 4
5 TO SORT ON MARKS 5
6 TO SORT ON TOTAL MARKS
7 TO SORT ON INDEX
8 TO SORT ON FIRST NAME
9 TO SORT ON LAST NAME
10 TO CREATE A OUTPUT FILE
11 TO EXIT THE PROGRAM AND FREE MEMORY

ENTER YOUR CHOICE----->
```

CREATE LINKED LIST:

In this module we read the data (i.e. S.No, Name of the student (i.e. First name and Last name) and marks of 5 subjects) from a (.txt) file.

The text file has the following format.

Index	First name	Middle name	Last name	Marks 1	Marks 2	Marks 3	Marks 4	Marks 5
1	Abhijeet	Sanjiv	Bonde	21	23	34	56	87

2	Damon	Joseph	Salvatore	43	56	76	89	09
3	Stefan	Joseph	Salvatore	27	54	56	52	56
:	:	:	:	:	:	:	:	:
99	Elena	David	Gilbert	85	59	32	58	65
100	Hemangi	Kunal	Sharma	34	65	82	45	98

For testing purpose, 100 students record have been processed but as such there is not limit on the number of students. As input data is taken through a file the same software can be used for generated the merit list of different classes.

The software uses linked list for efficient implementation. Therefore, a structure is required to implement the linked list containing the record of each student. The structure is declared as follows.

```
struct list
{
   int index;
   char f_name[20];
   char m_name[20];
   char l_name[20];
   int m1;
   int m2;
   int m3;
   int m4;
   int m5;
   int total_marks;
   float percentage;
   struct list *next;
};
```

At the start of this module, the module asks the user for the file name of the input file (as shown in Fig 1.1). If the program is not able to find a file with the given name it will show an error (as shown in Fig 1.2). So, the input file is user desired. The text file is then opened using FILE I/O. Once the file is opened successfully, data from the file is read and then stored in the structure which has been shown above. Two additional fields, total marks and percentage, are computed and added to the data while placing the records in the structure. Then the nodes of the structure are linked and thus the linked list is created. Once the linked list has been created then the file is closed for further simplification. Now all the student's records are present in the linked list and the same is referred for further processing.

Fig 1.1

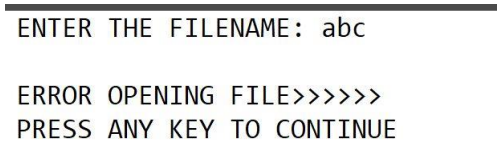

Fig 1.2

SORTING:

In this process the created linked list is sorted accrding to the the users input. User has options whether to sort on marks 1, marks 2, marks 3, marks 4, marks 5, total marks, First name, Last name, and index. In this process the linked list is not sorted but the index will be rearranged. Sorting in here is of 2 types:

1. Sorting based on Integer
2. Sorting based on String

The process of sorting based on Integer is as follows.

Let us assume that the user has selected the option to sort on marks 1. For the sorting to be done two arrays have been created (ie. Index and marks). The sorting is done by comparing the elements in the marks array. From the linked list that has been created before, the index and marks data has been copied in these arrays, respectively. From the field m1 of the linked list the data has been copied in the marks array. Once all the marks are in the marks array, the number of elements in the marks array are counted and then the sorting process starts. In this sorting process Bubble sort has been used to sort the index array. The sorting code is as follows.

```
for(i=0; i<(no_of_elements-1); i++)
{
   for(j=0; j<(no_of_elements-i-1); j++)
   {
      if(marks[j]<marks[j+1])
      {
         temp=marks[j];
         marks[j]=marks[j+1];
         marks[j+1]=temp;

         temp=index[j];
         index[j]=index[j+1];
         index[j+1]=temp;
      }
   }
}
```

The sorting is done by comparing the (j)th and (j+1)th elements from the marks array. If the (j)th element is smaller than (j+1)th element then (j)th and (j+1)th elements are interchanged and simultaneously the (j)th and (j+1)th elements of index array are also interchanged, as it will be required for preparing the List (Note that no change has been made in the linked list. The linked list remains the same as it was when created). The new

modified index which contains the index values as per the sorted merit list is then passed to the display function and to the Create output file function for the further process.

The process of sorting on string is as follows:
At first, a loop counts the number of elements in the list. A dummy index is created with its each element being (i+1) value, where i is the current position in the index array. Once the index array is filled, the strings (i.e. The first name or the last name) are compared with the help of ***strcmp()*** function (as the function returns, a negative value if the first string precedes the second string alphabetically, a value of zero if the first string and the second string are identical disregarding the case, a positive value if the second string precedes the first string alphabetically). If the output of the ***strcmp()*** function is positive then the elements are interchanged and simultaneously are the index values changed. At last the index is returned from the sort routine.

DISPLAY:

In this module the list is printed on the screen as per the index passed to it. In our case the index passed is sorted in ascending order. There are 2 parameters passed to this module (i.e. starting address of the Linked list and the index). With the index number that comes first, the pointer traverses through the linked list to print the record with the associated index. After printing each record, the pointer is again initialized to starting position of the linked list.

CREATE OUTPUT FILE:

In this module, the starting address of the linked list and index are the input parameters. It will ask the user to type the name of the file that the user wants to create for storing the result. Then the number of elements are counted according to which the data will be

written in the file. The data is written in a systematic manner, as there are proper functions used for the data writing process. The data is written in tabular form using the formatted I/O functions. The code segment which writes the data is as shown in Fig. 1.3.

```
do
{
    if(record->index == *(index+i))  // PRINTING THE RECORD AT I'TH INDEX
    {
        fprintf(fp, "%4d   ", record->index);
        fprintf(fp, "%-20s ", record->f_name);
        fprintf(fp, "%-20s ", record->m_name);
        fprintf(fp, "%-20s ", record->l_name);
        fprintf(fp, "%-8d ", record->m1);
        fprintf(fp, "%-8d ", record->m2);
        fprintf(fp, "%-8d ", record->m3);
        fprintf(fp, "%-8d ", record->m4);
        fprintf(fp, "%-8d ", record->m5);
        fprintf(fp, "%-5d ", record->total_marks);
        fprintf(fp, "%-10.3f ", record->percentage);
        fprintf(fp, "\n");
        break;
    }
    record = record->next;
}while(record!= NULL);
```

Fig. 1.3

FREE MEMORY AND EXIT:

This module is used to deallocate the memory that was reserved for storing the linked list. Once a node is freed the module prints a message onto the screen saying “FREED NODE NO: x” (where x is the node number). At last when the list pointer is pointing at null the program will end and will lead its way to exit.

COMPARISION BETWEEN INPUT AND OUTPUT:

- INPUT:

```
1 Abhijeet Sanjiv Bonde 21 23 34 56 87
2 Damon Joseph Salvatore 43 56 76 89 09
3 Stefan Joseph Salvatore 27 54 56 52 59
4 Lewis Damon Joushwa 32 43 14 61 82
5 Leon Albert Caverli 56 48 76 48 74
6 Abdul John Zadhea 32 43 54 24 56
7 Aadhya Klaus Mani 48 20 59 02 48
8 Laurence Andy Lee 25 89 47 63 14
9 David Natty Ping 48 69 75 93 52
10 Joseph George Tau 7 9 6 85 47
11 Nikita Rajaram Ranjan 85 96 47 24 3
12 Usha Murlidhar Rajan 88 57 15 64 31
13 Deelip Manohar Chaubey 98 75 61 43 68
14 Nilkanth Madhukar Mehera 81 62 83 87 6
15 Saurabh Praladh Saini 97 27 29 5 6
16 Kanika Bhalchandra Patil 96 26 4 8 32
17 Amitabh Bhupesh Kapoor 83 64 58 54 97
18 Jitendra Ananda Mahajan 78 25 53 24 36
19 Dharmendra Akshat Chopde 48 25 19 77 64
20 Riya Vishnu Londhe 47 59 67 25 06
```

- OUTPUT (merit list prepared on marks 1):

records=100

S.NO	FIRST NAME	MIDDLE NAME	LAST NAME	MARKS 1	MARKS 2	MARKS 3	MARKS 4	MARKS 5	TOTAL	PERCENTAGE
69	Alok	Girish	Chauhan	100	78	36	49	10	273	54.600
13	Deelip	Manohar	Chaubey	98	75	61	43	68	345	69.000
15	Saurabh	Praladh	Saini	97	27	29	5	6	164	32.800
62	Preeti	Ankur	Deshpande	97	52	64	58	66	337	67.400
64	Shirley	Avinash	Kulkarni	97	15	85	74	37	308	61.600
16	Kanika	Bhalchandra	Patil	96	26	4	8	32	166	33.200
32	Piyush	Utsav	Rathi	96	74	85	41	52	348	69.600
42	Radhika	Jatin	Kansal	94	96	97	98	59	444	88.800
46	Anuraag	Rajendra	Dwivedi	94	95	96	22	33	340	68.000
57	Roshini	Narayan	Batra	94	25	14	89	62	284	56.800
92	Nupur	Kartik	Aggrawal	94	64	13	34	18	223	44.600
86	Praveen	Pratik	Das	93	15	62	48	64	282	56.400
12	Usha	Murlidhar	Rajan	88	57	15	64	31	255	51.000
73	Barbie	Aditya	Chouhaan	87	25	69	64	19	264	52.800
11	Nikita	Rajaram	Ranjan	85	96	47	24	3	255	51.000

For the sake of completeness the merit list on total marks of 100 students prepared by the developed software is given as below.

records=100

S.NO	FIRST NAME	MIDDLE NAME	LAST NAME	MARKS 1	MARKS 2	MARKS 3	MARKS 4	MARKS 5	TOTAL	PERCENTAGE
1	Abhijeet	Sanjiv	Bonde	21	23	34	56	87	221	44.200
2	Damon	Joseph	Salvatore	43	56	76	89	9	273	54.600
3	Stefan	Joseph	Salvatore	27	54	56	52	59	248	49.600
4	Lewis	Damon	Joushwa	32	43	14	61	82	232	46.400
5	Leon	Albert	Caverli	56	48	76	48	74	302	60.400
6	Abdul	John	Zadhea	32	43	54	24	56	209	41.800
7	Aadhya	Klaus	Mani	48	20	59	2	48	177	35.400
8	Laurence	Andy	Lee	25	89	47	63	14	238	47.600
9	David	Natty	Ping	48	69	75	93	52	337	67.400
10	Joseph	George	Tau	7	9	6	85	47	154	30.800
11	Nikita	Rajaram	Ranjan	85	96	47	24	3	255	51.000
12	Usha	Murlidhar	Rajan	88	57	15	64	31	255	51.000
13	Deelip	Manohar	Chaubey	98	75	61	43	68	345	69.000
14	Nilkanth	Madhukar	Mehera	81	62	83	87	6	319	63.800
15	Saurabh	Praladh	Saini	97	27	29	5	6	164	32.800
16	Kanika	Bhalchandra	Patil	96	26	4	8	32	166	33.200
17	Amitabh	Bhupesh	Kapoor	83	64	58	54	97	356	71.200
18	Jitendra	Ananda	Mahajan	78	25	53	24	36	216	43.200
19	Dharmendra	Akshat	Chopde	48	25	19	77	64	233	46.600
20	Riya	Vishnu	Londhe	47	59	67	25	6	204	40.800
21	Yash	Vedu	Chaudhari	2	4	40	65	11	122	24.400
22	Aniket	Animesh	Chonde	30	21	24	27	64	166	33.200
23	Kiran	Ulhaas	Gonde	14	57	64	68	10	213	42.600
24	Suraj	Govinda	Bhole	54	85	64	31	25	259	51.800
25	Vidit	Anubhav	Gupta	51	85	21	31	65	253	50.600
26	Harshal	Abhishek	Sharma	54	98	65	32	54	303	60.600
27	Manohar	Prabhakar	Saxena	44	15	25	2	20	106	21.200
28	Harsh	Avinash	Singh	15	45	65	32	15	172	34.400
29	Mukti	Ramchandra	Baliyaan	84	62	4	2	30	182	36.400
30	Digambar	Lakshman	Sodi	2	5	7	80	99	193	38.600
31	Janhavi	Geetesh	Aggrawal	54	87	62	52	14	269	53.800
32	Piyush	Utsav	Rathi	96	74	85	41	52	348	69.600
33	Piyushi	Gangadhar	Bansal	25	14	69	47	58	213	42.600
34	Rahul	Khushal	Verma	31	64	97	79	46	317	63.400
35	Vineet	Chinmay	Rane	13	46	79	96	94	328	65.600
36	Vineeta	Vivek	Dayma	21	78	96	52	41	288	57.600
37	Himangi	Vinod	Lakka	14	87	32	95	84	312	62.400
38	Rohit	Harshit	Meherotra	84	95	15	62	61	317	63.400
39	Rohan	Rahul	Kolhe	15	48	59	26	19	167	33.400
40	Deepak	Varun	Bansal	19	37	46	28	64	194	38.800
41	Deepika	Shaan	Hurne	9	73	82	64	94	322	64.400
42	Radhika	Jatin	Kansal	94	96	97	98	59	444	88.800
43	Popat	Animesh	Munde	2	4	60	68	61	195	39.000
44	Mohan	Anil	Srivastava	78	94	16	19	18	225	45.000
45	Dhanraj	Sunail	Pande	78	64	89	13	44	288	57.600
46	Anuraag	Rajendra	Dwivedi	94	95	96	22	33	340	68.000
47	Kedar	Amar	Trevedi	33	44	85	25	45	232	46.400
48	Tanaya	Aman	Chaturvedi	15	4	6	9	4	38	7.600

49	Tanay	Akbar	Kar	15	48	62	18	14	157	31.400
50	Tanmay	Anthony	Chawla	16	17	84	92	36	245	49.000
51	Hardik	Parag	Chakarborty	25	84	61	31	34	235	47.000
52	Avishek	Ramesh	Ghosh	84	92	15	31	45	267	53.400
53	Rutik	Suresh	Ganguly	84	95	13	4	6	202	40.400
54	Sunaina	Vikram	Shulka	14	3	6	84	5	112	22.400
55	Roshan	Prashant	Dhoni	84	12	1	5	50	152	30.400
56	Kalpana	Bharat	Ahujja	66	48	58	57	91	320	64.000
57	Roshini	Narayan	Batra	94	25	14	89	62	284	56.800
58	Poonam	Pradeep	Kadam	35	18	27	39	15	134	26.800
59	Milind	Samarth	Rai	19	37	4	9	12	81	16.200
60	Heemant	Keshav	Roy	24	6	8	74	25	137	27.400
61	Shaila	Anup	Saraswat	9	85	64	78	25	261	52.200
62	Preeti	Ankur	Deshpande	97	52	64	58	66	337	67.400
63	Kabir	Amol	Deshmukh	48	97	26	49	34	254	50.800
64	Shirley	Avinash	Kulkarni	97	15	85	74	37	308	61.600
65	Jaya	Prateek	Talbar	7	5	0	54	9	75	15.000
66	Rakha	Mayur	Bacchan	3	69	98	78	52	300	60.000
67	Sushma	Prakash	Laddha	18	64	7	35	0	124	24.800
68	Hemmma	Yeshwant	Advani	54	98	41	2	58	253	50.600
69	Alok	Girish	Chauhan	100	78	36	49	10	273	54.600
70	Sanjay	Anmol	Kumar	0	25	78	1	64	168	33.600
71	Abir	Kumar	Singh	54	59	65	15	23	216	43.200
72	Daniel	Neils	Paul	58	29	23	45	47	202	40.400
73	Barbie	Aditya	Chouhaan	87	25	69	64	19	264	52.800
74	Mrunali	Suresh	Kumar	2	42	82	96	9	231	46.200
75	Aahana	Vishal	Kumar	48	89	85	76	87	385	77.000
76	Aalia	Rushikesh	Makur	84	29	16	79	31	239	47.800
77	Aditi	Sarpanch	Subraynam	70	9	84	35	19	217	43.400
78	Simranjeet	singh	Khalsa	79	64	19	37	17	216	43.200
79	Komal	Anil	Biradhe	16	4	6	73	49	148	29.600
80	Aditya	Suresh	Kadam	79	8	86	34	9	216	43.200
81	Vinay	Sushil	Gonarkar	6	35	93	24	97	255	51.000
82	Nameera	Anas	Khan	79	76	13	64	65	297	59.400
83	Anusha	Manish	Makan	25	2	3	26	56	112	22.400
84	Abhay	Sanjay	Rangnani	26	5	62	5	25	123	24.600
85	Bonnie	Klaus	Llyod	79	98	78	69	49	373	74.600
86	Praveen	Pratik	Das	93	15	62	48	64	282	56.400
87	Ashwini	Shobhraj	Motewar	79	85	28	36	49	277	55.400
88	Priyanka	Tanishq	Mahapatra	49	23	49	79	13	213	42.600
89	Heena	Priyanshu	Shinde	79	18	14	2	50	163	32.600
90	Kriti	Suraj	Gupta	19	28	98	82	67	294	58.800
91	Kirti	Ashwin	Goyal	49	64	11	34	16	174	34.800
92	Nupur	Kartik	Aggrawal	94	64	13	34	18	223	44.600
93	Prasanna	Rishabh	Jain	54	78	32	67	21	252	50.400
94	Prasanjeet	Abhishek	Kansal	23	65	7	98	12	205	41.000
95	Ranjan	Subhash	Banarjee	12	46	87	35	8	188	37.600
96	Bubbly	Shubham	Bandopadhya	26	76	97	8	57	264	52.800
97	Bablu	Subashish	Updhaya	14	64	79	37	69	263	52.600
98	Radhika	Somesh	Aapte	14	54	87	64	32	251	50.200
99	Elena	David	Gilbert	85	59	32	58	65	299	59.800
100	Hemangi	Kunal	Sharma	34	65	82	45	98	324	64.800

PRESS ANY KEY TO CONTINUE

END