**Expense Manager in C programming**

**A PROJECT REPORT**

Submitted by

**Raghul.B.S[RA2111050010052]**

**Dheraj Kumar Vks[RA2111050010029]**

**Nevedhano L[RA2111050010048]**

**Sai Aravinth[RA2111050010025]**

***Under the guidance of***

##### **Dr. V.V. Ramalingam**

(Associate Professor,  Computer Science and Engineering)

##### **M. Eliazer**

(Assistant Professor,  Computer Science and Engineering)



S.R.M. University, Kattankulathur, Kancheepuram District

|  |  |  |
| --- | --- | --- |
| 1 | #include<stdio.h> |  |
| 2 | #include<stdlib.h> |  |
| 3 | #include<string.h> |  |
| 4 | #define N 50 |  |
| 5 | #define M 30 |  |
| 6 | **double** currentincome=0; |  |
| 7 | **double** currentexpense=0; |  |
| 8 |  |  |
| 9 | **struct** node |  |
| 10 | { |  |
| 11 | **char** date[M]; |  |
| 12 | **double** amount; |  |
| 13 | **char** category[N]; |  |
| 14 | **struct** node \*next; |  |
| 15 | }\*income=NULL,\*expense=NULL; |  |
| 16 |  |  |
| 17 | **struct** record |  |
| 18 | { |  |
| 19 | **double** x,y; |  |
| 20 | }\*point=NULL; |  |
| 21 |  |  |
| 22 |  |  |
| 23 | **void** create(**char** x[],**double** y,**char** z[],**struct** node \*\*temp); |  |
| 24 | **void** display(**int** a3); |  |
| 25 | **struct** node \*readnext(**struct** node \*ptr,FILE \*fpointer); |  |
| 26 | **void** writeincome(**struct** node \*ptr); |  |
| 27 | **void** writeexpense(**struct** node \*ptr); |  |
| 28 | **void** deleterecord(**struct** node \*ptr); |  |
| 29 | **struct** node \*readincome(**struct** node \*ptr); |  |
| 30 | **struct** node \*readexpense(**struct** node \*ptr); |  |
| 31 | **void** write(**struct** record \*point); |  |
| 32 | **struct** record \*readrecord(); |  |
| 33 |  |  |
| 34 |  |  |
| 35 | **int** main() |  |
| 36 | { |  |
| 37 | **int** option,value; |  |
| 38 |  |  |
| 39 | **double** b; |  |
| 40 | **char** c[N],a[M]; |  |
| 41 | **char** s1[15],s2[15],s3[15]; |  |
| 42 |  |  |
| 43 |  |  |
| 44 |  |  |
| 45 | **if**(fopen("Record.bin","rb")!=NULL) |  |
| 46 | { |  |
| 47 | point=readrecord(); |  |
| 48 | currentincome=point->x; |  |
| 49 | currentexpense=point->y; |  |
| 50 | } |  |
| 51 |  |  |
| 52 | **if**(fopen("myincome.bin","rb")!=NULL) |  |
| 53 | { |  |
| 54 | income=readincome(income); |  |
| 55 | } |  |
| 56 | **if**(fopen("myexpense.bin","rb")!=NULL) |  |
| 57 | { |  |
| 58 | expense=readexpense(expense); |  |
| 59 | } |  |
| 60 |  |  |
| 61 |  |  |
| 62 | **do**{ |  |
| 63 |  |  |
| 64 | printf(" \n | " ); |

1. printf(" | YOUR INCOME = %.2lf INR \n ", currentincome);
2. printf(" | YOUR EXPENSE = %.2lf INR \n ", currentexpense);
3. printf(" | YOUR BALANCE = %.2lf INR \n ", currentincome-currentexpense);

68 printf(" | \n");

1. printf("ENTER THE OPTION FROM THE BELOW \n\n");
2. printf("1.INSERT INCOME \n");
3. printf("2.INSERT EXPENSE \n");
4. printf("3.VIEW INCOME RECORD \n");
5. printf("4.VIEW EXPENSE RECORD \n");
6. printf("5.EXIT\n");
7. scanf("%d",&option);
8. printf("\n\n\n"); 77

78 **switch**(option) 79 {

80 **case** 1:

81 printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\* ADD INCOME \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

82 printf("Enter The Date(e.g day month year)\n"); 83 scanf("%s %s %s",s1,s2,s3);

1. strcpy(a,"");
2. strcat(a,s1);
3. strcat(a," ");
4. strcat(a,s2);
5. strcat(a," ");
6. strcat(a,s3);
7. printf("Enter The Amount\n");
8. scanf("%lf",&b);
9. printf("Enter the Category\n");
10. scanf("%s",c); 94

95

96

97

1. currentincome=currentincome+b;
2. create(a,b,c,&income);
3. writeincome(income); 101

# break;

1. **case** 2:

104 printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\* ADD EXPENSE \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

105 printf("Enter The Date(e.g day month year)\n"); 106

107 scanf("%s %s %s",s1,s2,s3);

1. strcpy(a,"");
2. strcat(a,s1);
3. strcat(a," ");
4. strcat(a,s2);
5. strcat(a," ");
6. strcat(a,s3); 114
7. printf("Enter The Amount\n");
8. scanf("%lf",&b);
9. printf("Enter The Category\n");
10. scanf("%s",c); 119

120

1. currentexpense=currentexpense+b;
2. create(a,b,c,&expense);
3. writeexpense(expense); 124

# break;

1. **case** 3:
2. printf("\*\*\*\*\*\*\*\*\* YOUR INCOME RECORD IS \*\*\*\*\*\*\*\n\n");
3. display(3);

# break;

1. **case** 4:
2. printf("\*\*\*\*\*\*\*\*\* YOUR EXPENSE RECORD IS \*\*\*\*\*\*\*\n\n");
3. display(4);

# break;

1. **case** 5:
2. point=(**struct** record\*)malloc(**sizeof**(**struct** record));
3. point->x=currentincome;
4. point->y=currentexpense;
5. write(point);

# break;

1. **default**:
2. printf("WRONG OPTION SELECTED -Enter Valid Option");

# break;

143 }

144 }**while**(option!=5); 145

146

147 **return** 0;

148 } 149

150 **void** create(**char** x[],**double** y,**char** z[],**struct** node \*\*temp)

151 {

1. **struct** node \*newnode,\*ptr;
2. newnode=(**struct** node\*)malloc(**sizeof**(**struct** node));
3. **if**(\*temp==NULL)

155 {

1. strcpy(newnode->date,x);
2. newnode->amount=y;
3. strcpy(newnode->category,z);
4. newnode->next=NULL;
5. \*temp=newnode;

161 }

162 **else**

163 {

1. ptr=\*temp;
2. **while**(ptr->next!=NULL)

166 {

167 ptr=ptr->next;

168 }

1. strcpy(newnode->date,x);
2. newnode->amount=y;
3. strcpy(newnode->category,z);
4. newnode->next=NULL;
5. ptr->next=newnode;

174 }

175 } 176

177 **void** deleterecord(**struct** node \*ptr)

178 {

1. **struct** node \*freeme =ptr;
2. **struct** node \*holdme=NULL;
3. **while**(freeme!=NULL)

182 {

1. holdme=freeme->next;
2. free(freeme);
3. freeme=holdme;

186 }

187 } 188

189 **struct** node \*readnext(**struct** node \*ptr,FILE \*fpointer)

190 { 191

192 **if**(ptr==NULL)

193 {

1. ptr=(**struct** node \*)malloc(**sizeof**(**struct** node));
2. fread(ptr,**sizeof**(**struct** node),1,fpointer);
3. ptr->next=NULL;

197 }

198 **else**

199 {

1. **struct** node \*ptr1=ptr;
2. **struct** node \*ptr2=(**struct** node \*)malloc(**sizeof**(**struct** node));
3. **while**(ptr1->next!=NULL)

203 {

204 ptr1=ptr1->next;

205 }

1. fread(ptr2,**sizeof**(**struct** node),1,fpointer);
2. ptr1->next=ptr2;
3. ptr2->next=NULL;

209 }

210 **return** ptr;

211 } 212

213 **struct** node \*readincome(**struct** node \*ptr)

214 {

1. FILE \*fpointer;
2. fpointer=fopen("myincome.bin","rb");
3. **if**(fpointer!=NULL)

218 {

1. deleterecord(ptr);
2. ptr=NULL;
3. fseek(fpointer,0,SEEK\_END);
4. **long** filesize=ftell(fpointer);
5. rewind(fpointer);
6. **int** entries=(**int**)(filesize/(**sizeof**(**struct** node)));
7. **for**(**int** i=0;i<entries;i++)

226 {

1. fseek(fpointer,(**sizeof**(**struct** node)\*i),SEEK\_SET);
2. ptr=readnext(ptr,fpointer);

229 }

230 }

231 **else**

232 {

233 printf("ERROR IN OPENINNG FILE\n");

234 }

235 **return** ptr;

236 } 237 238

239 **void** display(**int** a3)

240 {

241 **if**(a3==3)

242 {

243

244 **if**(fopen("myincome.bin","rb")==NULL)

245 {

1. printf("NO RECORDS AVAILABLE\n\n");
2. printf(

"

\_\n\n"); 248

|  |  |  |
| --- | --- | --- |
| 249 |  | } |
| 250 |  | **else** |
| 251 | { |  |
| 252 |  |  |
| 253 |  | **struct** node \*ptr2=income; |
| 254 |  | **while**(ptr2!=NULL) |
| 255 |  | { |
| 256 |  | printf("Date: %s\nAmount: %.2lf INR\nCategory: %s\n\n",ptr2->date,ptr2-> |
| 257 |  | amount,ptr2->category); |
| 258 |  | ptr2=ptr2->next; |
| 259 |  | } |

260 printf(

"

\_\n\n"); 261

262 }

263 }

264 **else if**(a3==4)

265 {

266

267 **if**(fopen("myexpense.bin","rb")==NULL)

268 {

1. printf("NO RECORDS AVAILABLE\n\n");
2. printf(

"

\_\n\n");

271 }

272 **else**

273 {

274

1. // expense=readexpense(expense);
2. **struct** node \*ptr2=expense;
3. **while**(ptr2!=NULL)

278 {

1. printf("Date: %s\nAmount: %.2lf INR\nCategory: %s\n\n",ptr2->date, ptr2->
2. amount,ptr2->category);
3. ptr2=ptr2->next;

282 }

283 printf(

"

\_\n\n"); 284

285

286 }

287

288 } 289

290 } 291

292 **void** writeincome(**struct** node \*ptr)

293 {

1. FILE \*fpointer;
2. fpointer=fopen("myincome.bin","wb");
3. **if**(fpointer!=NULL)

297 {

1. **struct** node \*ptr1=ptr;
2. **struct** node \*holdnext=NULL;
3. **while**(ptr1!=NULL)

301 {

1. holdnext=ptr1->next;
2. ptr1->next=NULL;
3. fseek(fpointer,0,SEEK\_END);
4. fwrite(ptr1,**sizeof**(**struct** node),1,fpointer);
5. ptr1->next=holdnext;
6. holdnext=NULL;
7. ptr1=ptr1->next;

309 }

1. fclose(fpointer);
2. fpointer=NULL;
3. printf("\nINCOME SAVED SUCCESSFULLY\n\n");
4. printf(

"

\_\n\n"); 314

315 }

# else{

1. printf("\nCANNOT SAVE INCOME..TRY AGAIN\n");
2. printf(

"

\_\n\n"); 319

320 }

321 } 322 323 324 325 326

327 **void** writeexpense(**struct** node \*ptr)

328 {

1. FILE \*fpointer;
2. fpointer=fopen("myexpense.bin","wb");
3. **if**(fpointer!=NULL)

332 {

1. **struct** node \*ptr1=ptr;
2. **struct** node \*holdnext=NULL;
3. **while**(ptr1!=NULL)

336 {

1. holdnext=ptr1->next;
2. ptr1->next=NULL;
3. fseek(fpointer,0,SEEK\_END);
4. fwrite(ptr1,**sizeof**(**struct** node),1,fpointer);
5. ptr1->next=holdnext;
6. holdnext=NULL;
7. ptr1=ptr1->next;

344 }

1. fclose(fpointer);
2. fpointer=NULL;
3. printf("\nEXPENSE SAVED SUCCESSFULLY\n\n");
4. printf(

"

\_\n\n"); 349

350 }

# else{

1. printf("\nCANNOT SAVE EXPENSE..TRY AGAIN\n\n");
2. printf(

"

\_\n\n"); 354

355 }

356 } 357 358

359 **struct** node \*readexpense(**struct** node \*ptr)

360 {

1. FILE \*fpointer;
2. fpointer=fopen("myexpense.bin","rb");
3. **if**(fpointer!=NULL)

364 {

1. deleterecord(ptr);
2. ptr=NULL;
3. fseek(fpointer,0,SEEK\_END);
4. **long** filesize=ftell(fpointer);
5. rewind(fpointer);
6. **int** entries=(**int**)(filesize/(**sizeof**(**struct** node)));
7. **for**(**int** i=0;i<entries;i++)

372 {

1. fseek(fpointer,(**sizeof**(**struct** node)\*i),SEEK\_SET);
2. ptr=readnext(ptr,fpointer);

375 }

376 }

377 **else**

378 {

379 printf("cannonot open file\n"); 380

381 }

382 **return** ptr;

383 } 384

385 **void** write(**struct** record \*point)

386 {

1. FILE \*fpointer;
2. fpointer=fopen("Record.bin","wb");
3. **if**(fpointer!=NULL)

390 { 391

1. fseek(fpointer,0,SEEK\_END);
2. fwrite(point,**sizeof**(**struct** record),1,fpointer);}

# else{

1. printf("FILEOPEN ERROR\n");

396 }

1. fclose(fpointer);
2. fpointer=NULL; 399

400 } 401

402 **struct** record \*readrecord()

403 {

404 FILE \*fpointer;

405 fpointer=fopen("Record.bin","rb");

406 **struct** record \*ptr=NULL; 407

408 **if**(fpointer!=NULL)

409 { 410

411 fseek(fpointer,0,SEEK\_SET); 412

413 ptr=(**struct** record \*)malloc(**sizeof**(**struct** record));

414 fread(ptr,**sizeof**(**struct** record),1,fpointer); 415

416

417 }

418 **else**

419 {

420 printf("CANNOT OPEN FILE\n");

421 }

422 **return** ptr;

423 }











