Raed Alsheikh Amin

Algorithm Analysis

readTransactions Function

Code	Cost	Times
void readTransactions (FILE* inFile, customer* customers, int no_of_customers) {		
char c,tempname[8],faketoskip[100];	c1	1
int countlines=0,i=0,j=0,uniquecust=0,NumberOfItemsPurchased=0;	c2	1
float CostPerItem=0.0,amountPaid=0.0;	c3	1
inFile=fopen("transactions.txt","r");	c4	1
if (inFile == NULL){	c5	1
printf("File could not be opened.\n");		
exit(1);		
}		
while ((c = fgetc(inFile)) != EOF) {//fgetc stops when it counters end of file(EOF)	c6	m+1
// Increment the count when a newline character is encountered		
if (c == '\n') {	c7	n+1
countlines++;	c8	n
}		
}		
rewind(inFile);//to get the crosser to the beginning of the file.	c9	1
//to skip the first line of the file		
while((c=fgetc(inFile))!=EOF){	c10	k+1
if(c =='\n')//we consider only the first new line	c11	k

break;	c12	1
}		
for(i=0;i<countlines;i++)	c13	n+1
{		
fscanf(inFile,"%[^;];%[^;];%[^;];%[^;];%d;%f;%*[^\n]\n",tempname,faketoskip,faketoskip,faketoskip,&NumberOfItemsPurchased,&CostPerItem);	c14	n
amountPaid=NumberOfItemsPurchased*CostPerItem;	c15	n
int isDuplicate = 0;	c16	n
for (j = 0; j < uniquecust; j++) {	c17	n(h+1)
if (strcmp(customers[j].name, tempname) == 0) {	c18	n*h
isDuplicate = 1;	c19	O(n*h)
customers[j].transactions++;	c20	O(n*h)
customers[j].items_purchased+=NumberOfItemsPurchased;	c21	O(n*h)
customers[j].amount_paid+=amountPaid;	c22	O(n*h)
break;	c23	O(n*h)
}		
}		
if (!isDuplicate) {//this will be done for one time for each customer.	C24	n*h
strcpy(customers[uniquecust].name, tempname);	c25	h
customers[uniquecust].transactions=1;	c26	h
customers[uniquecust].items_purchased=NumberOfItemsPurchased;	c27	h
customers[uniquecust].amount_paid=amountPaid;	c28	h
uniquecust++;	c29	h
}		
}		
fclose(inFile); }	c30	1

m is the number of characters in the txt file.

n is the number of new lines, which means customers’ transictions.

K is the number of characters in one single line.

h is the number of unique customers

According to the table of Cost and Time, the worst case happens when all the customers are unique.

- ⇨ h=n =>

Total cost = O(n*n) = O(n^2)