CS143 – 7TH PROJECT

Retail analysis program

MARK THARWAT SAMIR TAWFIK
19200164

Supervised by Mohamed Elhabrouk & Nagy K. Aly

TABLE OF CONTENTS

CS143 – 7TH PROJECT

1. Introduction

The program tends to be a retail company retail analysis software that stores information about the performance of the company's' branches. For each branch, the program stores the sales revenue over the course of 12 months related to each branch.

2. Functions of the program

1. *Allows the user to enter the sales data for all branches and for all months and store them in single data structure.*
2. *Allows user to add a new branch or delete an existing branch record*
3. *Calculate the total sales of the company.*
4. *Calculate the percentage share of each branch in the total sales.*
5. *Determine the month of the peak sales.*
6. *Allows the user to specify a certain month, and in response the program displays the list of branch indices sorted in a descending order based on the sales revenue.*
7. *Allows the user to specify a certain branch, and in response the program displays the list of months indices sorted in a descending order based on the sales revenue.*
8. *Allows the user to choose the functionality he wishes to make use of. The user should be prompted with a question if he wishes to continue or not. The program should terminate if the user does not want to continue.*

3. DATA STRUCTURE

Data Structure in C Programming Language is a specialized format for organizing and storing data. In General data structure types include the file, array, record, table, tree.. etc.

- **Array:** Array is collection of similar data type, you can insert and deleted element form array without follow any order.
- **Stack:** Stack work on the basis of Last-In-First-Out (LIFO). Last entered element removed first.
- **Queue:** Queue work on the basis of First-In-First-Out (FIFO). First entered element removed first.
- **Linked List:** Linked list is the collection of node, Here you can insert and delete data in any order.
- **Tree:** Stores data in a non linear form with one root node and sub nodes.

The data structure used in the program is Array for the following reasons:

- collection of similar types of data.

  For the fact that the program only stores numbers.

- we have to only remember the first index of array.

  The first element address in the array acts as a pointer to the array.

- used to implement other data structure like linked lists,stack,queue, trees, graph etc.

  In case that we want to change the data structure in the future then Array is the best base.

- 2 Dimensional array is used to represent a matrix.

  In the program the data will be stored matrix like, 2D array would be a very convenient data structure.

3. Program functions

3.1 Main function

```
int main(int argc, const char * argv[]) {
   int countOfBranches;
   printf("*****Welcome to the retail analysis app*****\n");
   printf("Kindly specify the count of branches: ");
   scanf("%d",&countOfBranches);
   //creating the data matrix (2-D Array)
   float mainDataMatrix[countOfBranches+99][12]; //12 for a complete year
   //***********************************************************************************************//
   //filling the matrix
   for (int i=0; i<countOfBranches; i++) {
      printf("Sales for branch %d:\n",i+1);
      for (int j=0; j<12; j++) {
         printf("sales for month %d:",j+1);
         scanf("%f",&mainDataMatrix[i][j]);
      }
      printf("\n");
   }
   //***********************************************************************************************//
   //prompt menu
   int userChoice;
   float totalSales;
   do {
      MainMenuOptions();
      scanf("%d",&userChoice);
      switch (userChoice) {
         case 1:
            MatrixPrint(mainDataMatrix, countOfBranches);
            break;
         case 2:
            MatrixAddition(mainDataMatrix, countOfBranches);
            MatrixPrint(mainDataMatrix, countOfBranches+1);
            countOfBranches+=1; //this is an addition function
            break;
         case 3:
            printf("\n[Information!]Current count of branches is: %d\n", countOfBranches);
            MatrixDeletion(mainDataMatrix, &countOfBranches);
            MatrixPrint(mainDataMatrix, countOfBranches);
```

The main function is where the program starts, the first thing it does is to ask the user for the count of the available branches to be inserted. Then another function called MainMenu is being called, so it previews the options available and the functions that the application can do. Then another do while loop begins that keeps showing the menu each time a process is finished or wrong input was inserted from the user. Inside do while loop there is a switch case that changes the flow of the program based on the selection of the user.

3.2 MainMenu function

```
void MainMenuOptions(){
printf("**-----------------------------------**\n");
printf("**Please pick of the following options**\n");
printf("**-----------------------------------**\n");
printf("1. Preview the Branch\\Sales matrix\n");
printf("2. Add a new branch\n");
printf("3. Delete an existing branch\n");
printf("4. Calculate total sales\n");
printf("5. Calculate percentage share of each branch\n");
```

The MainMenu function prints the options that the user can pick from using the application on the main menu.

3.3 MatrixPrint function

```
void MatrixPrint(float arr[][12], int countOfBranches){
   printf("Branch\\Month:");//header of the table
   for (int i=0; i<12; i++) {//header of the table (Cont.)
      printf("%9d",i+1);
   }
   printf("\n");
   for(int x=0;x<countOfBranches;x++){
      printf("Branch %d:        ",x+1);
      for (int y=0; y<12; y++) {
```

MatrixPrint function takes two inputs which are the address of the array to print and the count of branches to know what should be printed from the data in the program, as through the program the countOfBranches is the controller, as it helps each function identifying the current active data that should be handled.

3.4 MatrixAddition

```
void MatrixAddition(float arr[][12], int countOfBranches){
   for(int i=0;i<12;i++){
      printf("sales for month %d:",i+1);
      scanf("%f",&arr[countOfBranches][i]);
```

Similar to MatrixPrint this function takes two inputs which are the address of the array to print and the count of branches to know what should be used from the data in the program, as through the program the countOfBranches is the controller, as it helps each function identifying the current active data that should be handled. MatrixAddition adds a new branch by adding new 12 elements to the array. On the other hand after calling MatrixAddition the main function increments the count of branches by 1 to let the whole program knows that the count of branches has been increased.

3.5 MatrixDeletion

```
void MatrixDeletion(float arr[][12], int *countOfBranches){
    int oldCountOfBranches = *countOfBranches;
    int newCountOfBranches;
    int targetToBeDeleted;
    printf("Input the number of Branch you want to delete: ");
    scanf("%d",&targetToBeDeleted);
    if (oldCountOfBranches==1) { //only one branch data exists
        printf("[Warning!]You have cleared the app data\n");
        newCountOfBranches=0;
        *countOfBranches=newCountOfBranches;
    }else if (oldCountOfBranches==2){ //we can only shrink the array in this case, he wants to delete the second row
        if (targetToBeDeleted==2) {
            newCountOfBranches=1;
            *countOfBranches=newCountOfBranches;
        }else{//he wants to delete the first row
            for (int i=0; i<12; i++) {//we shift the second cell up
                arr[0][i] = arr[1][i];
            }
            newCountOfBranches=1;
            *countOfBranches=newCountOfBranches;
        }
    }else if (oldCountOfBranches>2){ //we have more than 2 branches (swap mechanics)
        if (targetToBeDeleted==1) { //he wants to delete the first row
            for (int i=0; i<12; i++) {//we shift the second cell up
                arr[0][i] = arr[oldCountOfBranches-1][i];
```

MatrixDeletion takes two inputs which are the address of the array to print and the count of branches to know what should be used from the data in the program, as through the program the countOfBranches is the controller, as it helps each function identifying the current active data that should be handled.

MatrixDeletion has 3 different scenarios. The first is the existence of only one branch in the matrix, in this case the countOfBranches is decremented to 0 and this is being announced to the whole program through a pointer. The second case is when there is two branches in the matrix, in this case we only shrink the matrix by decrementing countOfBranches in case that the second line is to be deleted and if the user aims to delete the first line the program swap the two branches data and then shrink the matrix with the same technique. The last case is whent the count of branches is greater than 2 and, in this case, we use swap and shrink technique that was explained ealier in this paragraph.

3.6 TotalSales function

```
float TotalSales(float arr[][12],int countOfBranches){
   float totalSales=0;
   for (int i=0; i<countOfBranches; i++) {
      for (int j=0; j<12; j++) {
         totalSales+=arr[i][j];
      }
```

TotalSales function takes two inputs which are the address of the array to print and the count of branches to know what should be used from the data in the program, as through the program the countOfBranches is the controller, as it helps each function identifying the current active data that should be handled. The function goes through all the elements in the matrix up to the countOfBrnaches limit and then returns the total sales value to the main function for printing.

3.7 SalesShare function

```
void SalesShare(float arr[][12],int countOfBranches,int totalSales){
   for (int i=0; i<countOfBranches; i++) {
      float sum=0;
      for (int j=0; j<12; j++) {
         sum+=arr[i][j];
      }
```

SalesShare function takes two inputs which are the address of the array to print and the count of branches to know what should be used from the data in the program, as through the program the countOfBranches is the controller, as it helps each function identifying the current active data that should be handled. It goes though the branch row sales and calculates the sum of the sales then prints the percentage of the sales towards the total sales that is being returned from TotalSales function.

3.8 PeakSales function

```
void PeakSales(float arr[][12],int countOfBranches){
   float peak=0;
   int monthOfPeak=0;
   for (int i=0; i<12; i++) {
      int sum=0;
      for (int j=0; j<countOfBranches; j++) {
         sum+=arr[j][i];
         if (peak<sum) {
            peak = sum;
            monthOfPeak = i+1;
```

PeakSales function takes two inputs which are the address of the array to print and the count of branches to know what should be used from the data in the program, as through the program the countOfBranches is the controller, as it helps each function identifying the current active data that should be handled. It searches through the months in the programs (column) looking for the greatest value and relate it to the corresponding month value, then print it.

3.9 MonthSales function

```
void MonthSales(float arr[][12],int countOfBranches){
  int targetMonth;
  printf("Which month? ");
  scanf("%d",&targetMonth);
  float tempArray[countOfBranches]; //array of branches sales
  float tempArrayPositions[countOfBranches]; //array for positions
  for (int i=1; i<=countOfBranches; i++) {
    tempArrayPositions[i-1]=i;
  }
  for (int i=0; i<countOfBranches; i++) {
    tempArray[i]=arr[i][targetMonth-1];
  }
  for(int i=1;i<countOfBranches;i++){ //passes
    for(int j=0;j<countOfBranches-i;j++){ //positions
      if(tempArray[j]<tempArray[j+1]){
        int temp = tempArray[j]; //sorting the sales values
        tempArray[j] = tempArray[j+1];
        tempArray[j+1] = temp;
```

MonthSales function takes two inputs which are the address of the array to print and the count of branches to know what should be used from the data in the program, as through the program the countOfBranches is the controller, as it helps each function identifying the current active data that should be handled. It creates two sub functions. The first one is tempArray and the second one is tempArrayPositions, the first arrat holds the selected month data and then sort the array using the bubble sort algorithm, the tempArrayPositions receives the same changes happening in in temoArray so we can have a sorted positions and value as an output.

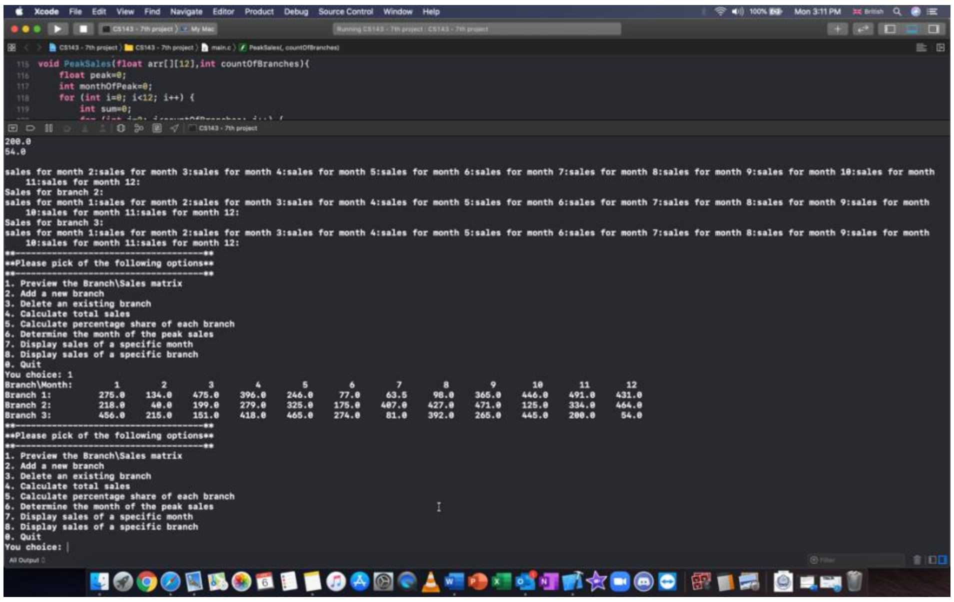
3.10 BranchSales function

```
void BranchSales(float arr[][12],int countOfBranches){
  int targetBranch;
  printf("Which branch? ");
  scanf("%d",&targetBranch);
  float salesValues[12];
  float salesValuesPositions[12];

  for (int i=0; i<12; i++) {
    salesValuesPositions[i]=i+1;
  }

  for (int i=0; i<12; i++) {
    salesValues[i]=arr[targetBranch-1][i];
  }

  for (int x=0; x<11; x++) {
    int indexOfmax=x;
    for (int y=x+1; y<12; y++) {
      if (salesValues[y]>salesValues[indexOfmax]) {
        indexOfmax=y;
      }
    }
```

BranchSales function takes two inputs which are the address of the array to print and the count of branches to know what should be used from the data in the program, as through the program the countOfBranches is the controller, as it helps each function identifying the current active data that should be handled. It follows the same technique that MonthSales follows as it creates sub array with the data to be sorted and apply the selection sort algorithm on these sub arrays to reach the required order.

4. PROGRAM SCREENSHOTS

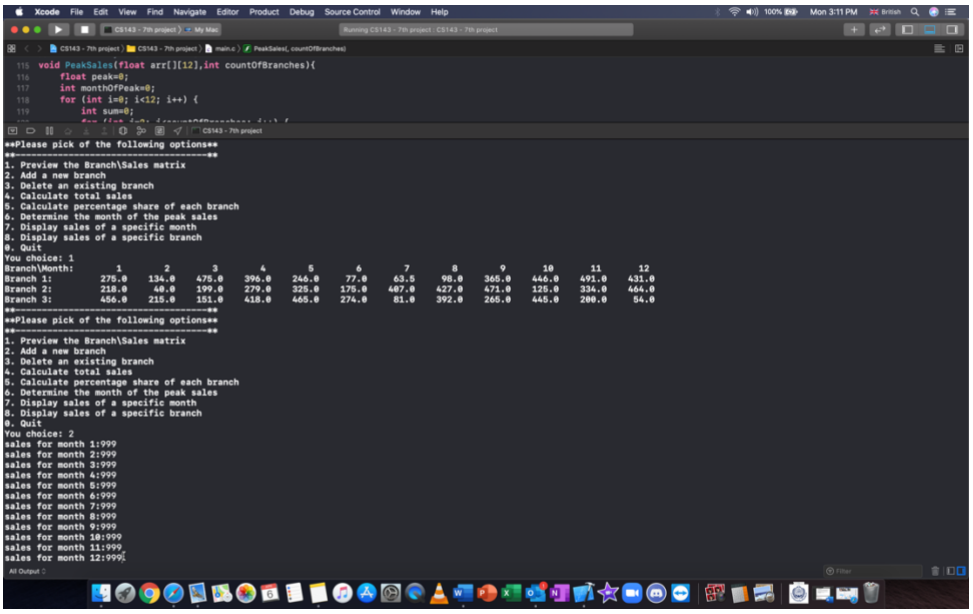
4.1 INPUT AND PRINT


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
CS143 - 7th project › My Mac
Running CS143 - 7th project : CS143 - 7th project
CS143 - 7th project › CS143 - 7th project › main.c › PeakSales(, countOfBranches)
All Output

```c
void PeakSales(float arr[][12],int countOfBranches){
    float peak=0;
    int monthOfPeak=0;
    for (int i=0; i<12; i++) {
        int sum=0;
```

```
200.0
54.0

sales for month 2:sales for month 3:sales for month 4:sales for month 5:sales for month 6:sales for month 7:sales for month 8:sales for month 9:sales for month 10:sales for month
    11:sales for month 12:
Sales for branch 2:
sales for month 1:sales for month 2:sales for month 3:sales for month 4:sales for month 5:sales for month 6:sales for month 7:sales for month 8:sales for month 9:sales for month
    10:sales for month 11:sales for month 12:
Sales for branch 3:
sales for month 1:sales for month 2:sales for month 3:sales for month 4:sales for month 5:sales for month 6:sales for month 7:sales for month 8:sales for month 9:sales for month
    10:sales for month 11:sales for month 12:
**----------------------------------**
**Please pick of the following options**
**----------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: 1
Branch\Month:        1       2       3       4       5       6       7       8       9      10      11      12
Branch 1:        275.0   134.0   475.0   396.0   246.0    77.0    63.5    98.0   365.0   446.0   491.0   431.0
Branch 2:        218.0    40.0   199.0   279.0   325.0   175.0   407.0   427.0   471.0   125.0   334.0   464.0
Branch 3:        456.0   215.0   151.0   418.0   465.0   274.0    81.0   392.0   265.0   445.0   200.0    54.0
**----------------------------------**
**Please pick of the following options**
**----------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: 
```

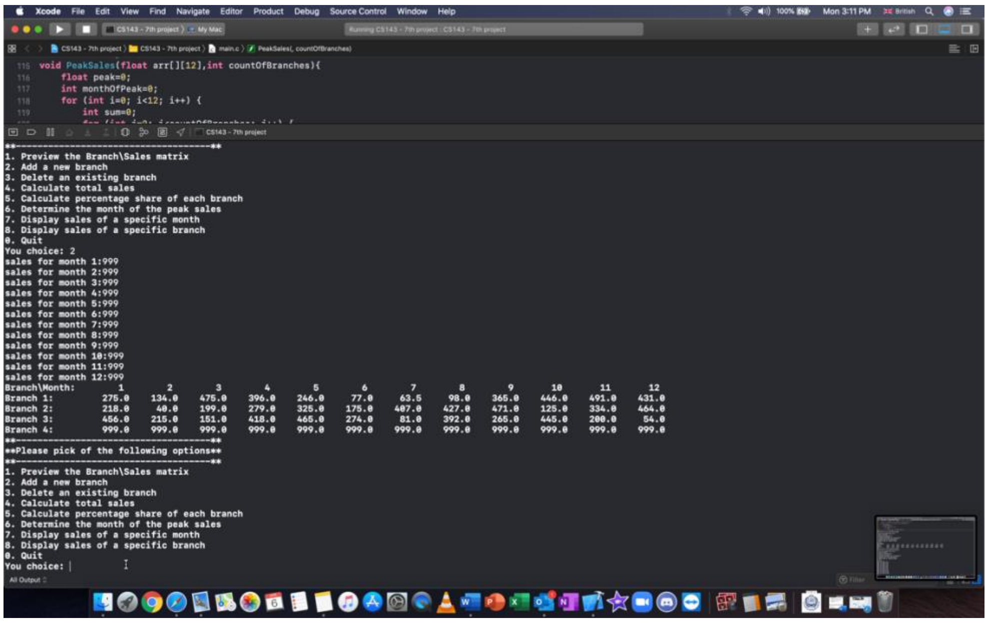
4.2 ADDING NEW BRANCH


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
CS143 - 7th project › My Mac
Running CS143 - 7th project : CS143 - 7th project
CS143 - 7th project › CS143 - 7th project › main.c › PeakSales(, countOfBranches)
All Output

```c
void PeakSales(float arr[][12],int countOfBranches){
    float peak=0;
    int monthOfPeak=0;
    for (int i=0; i<12; i++) {
        int sum=0;
```

```
**Please pick of the following options**
**----------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: 1
Branch\Month:        1       2       3       4       5       6       7       8       9      10      11      12
Branch 1:        275.0   134.0   475.0   396.0   246.0    77.0    63.5    98.0   365.0   446.0   491.0   431.0
Branch 2:        218.0    40.0   199.0   279.0   325.0   175.0   407.0   427.0   471.0   125.0   334.0   464.0
Branch 3:        456.0   215.0   151.0   418.0   465.0   274.0    81.0   392.0   265.0   445.0   200.0    54.0
**----------------------------------**
**Please pick of the following options**
**----------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: 2
sales for month 1:999
sales for month 2:999
sales for month 3:999
sales for month 4:999
sales for month 5:999
sales for month 6:999
sales for month 7:999
sales for month 8:999
sales for month 9:999
sales for month 10:999
sales for month 11:999
sales for month 12:999
```

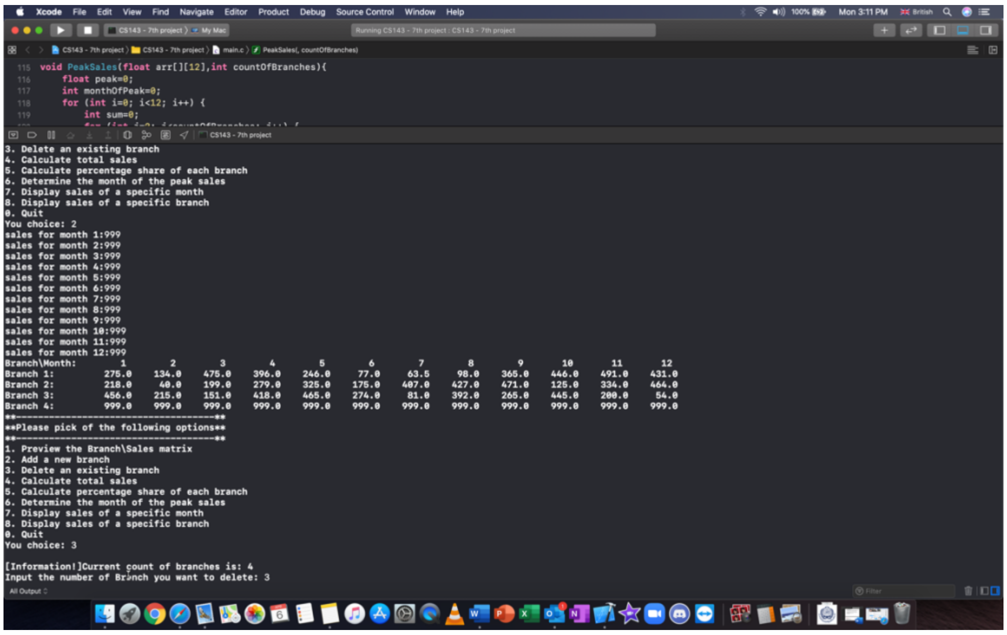
4.3 Addition complete



```
void PeakSales(float arr[][12],int countOfBranches){
    float peak=0;
    int monthOfPeak=0;
    for (int i=0; i<12; i++) {
        int sum=0;
```

```
**-------------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: 2
sales for month 1:999
sales for month 2:999
sales for month 3:999
sales for month 4:999
sales for month 5:999
sales for month 6:999
sales for month 7:999
sales for month 8:999
sales for month 9:999
sales for month 10:999
sales for month 11:999
sales for month 12:999
Branch\Month:       1       2       3       4       5       6       7       8       9      10      11      12
Branch 1:       275.0   134.0   475.0   396.0   246.0    77.0    63.5    98.0   365.0   446.0   491.0   431.0
Branch 2:       218.0    40.0   199.0   279.0   325.0   175.0   407.0   427.0   471.0   125.0   334.0   464.0
Branch 3:       456.0   215.0   151.0   418.0   465.0   274.0    81.0   392.0   265.0   445.0   200.0    54.0
Branch 4:       999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0
**-------------------------------------**
**Please pick of the following options**
**-------------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: |
```

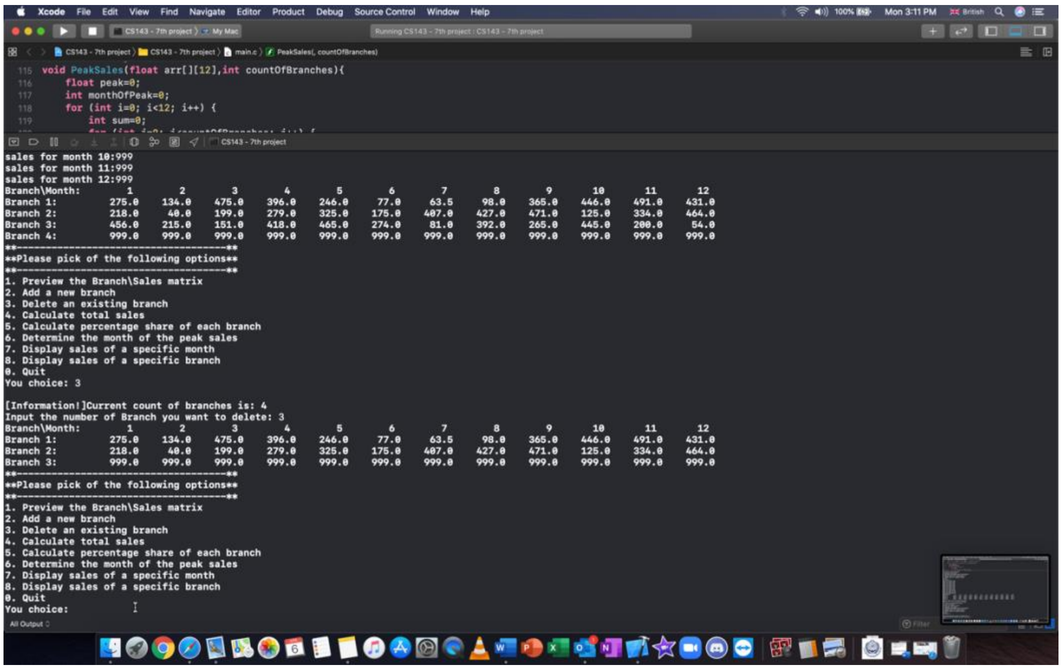
4.4 Using delete function



```
void PeakSales(float arr[][12],int countOfBranches){
    float peak=0;
    int monthOfPeak=0;
    for (int i=0; i<12; i++) {
        int sum=0;
```

```
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: 2
sales for month 1:999
sales for month 2:999
sales for month 3:999
sales for month 4:999
sales for month 5:999
sales for month 6:999
sales for month 7:999
sales for month 8:999
sales for month 9:999
sales for month 10:999
sales for month 11:999
sales for month 12:999
Branch\Month:       1       2       3       4       5       6       7       8       9      10      11      12
Branch 1:       275.0   134.0   475.0   396.0   246.0    77.0    63.5    98.0   365.0   446.0   491.0   431.0
Branch 2:       218.0    40.0   199.0   279.0   325.0   175.0   407.0   427.0   471.0   125.0   334.0   464.0
Branch 3:       456.0   215.0   151.0   418.0   465.0   274.0    81.0   392.0   265.0   445.0   200.0    54.0
Branch 4:       999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0
**-------------------------------------**
**Please pick of the following options**
**-------------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: 3

[Information!]Current count of branches is: 4
Input the number of Branch you want to delete: 3
```

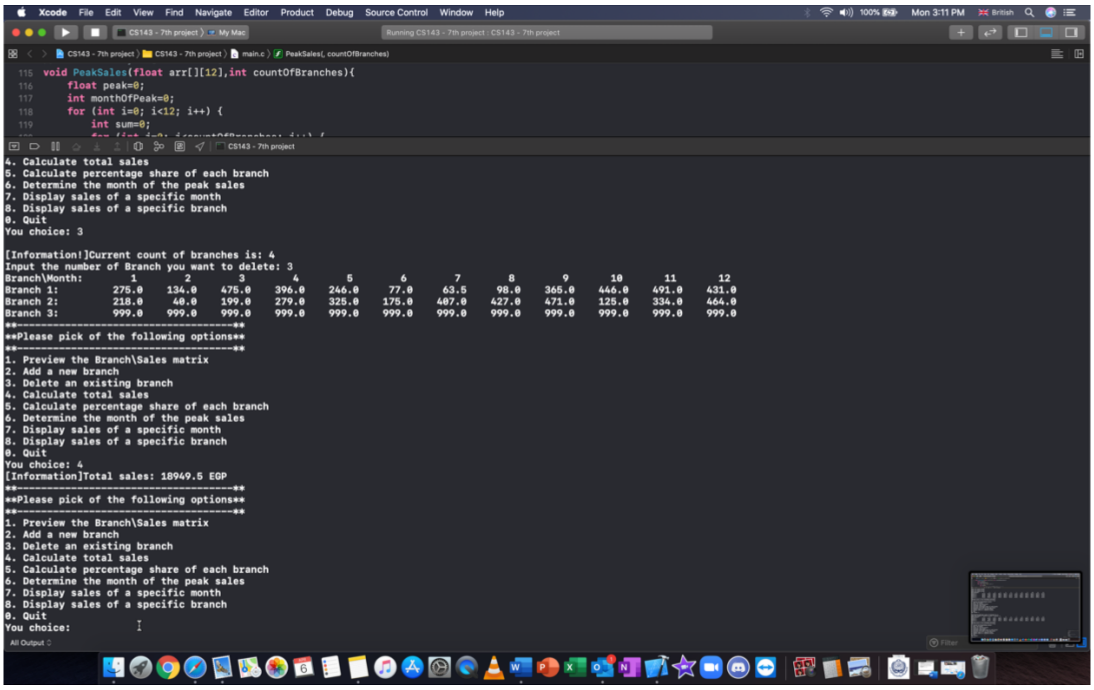
4.5 Delete function completed



```c
void PeakSales(float arr[][12],int countOfBranches){
    float peak=0;
    int monthOfPeak=0;
    for (int i=0; i<12; i++) {
        int sum=0;
```

```
sales for month 10:999
sales for month 11:999
sales for month 12:999
Branch\Month:        1       2       3       4       5       6       7       8       9      10      11      12
Branch 1:        275.0   134.0   475.0   396.0   246.0    77.0    63.5    98.0   365.0   446.0   491.0   431.0
Branch 2:        218.0    40.0   199.0   279.0   325.0   175.0   407.0   427.0   471.0   125.0   334.0   464.0
Branch 3:        456.0   215.0   151.0   418.0   465.0   274.0    81.0   392.0   265.0   445.0   200.0    54.0
Branch 4:        999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0
**------------------------------------**
**Please pick of the following options**
**------------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: 3

[Information!]Current count of branches is: 4
Input the number of Branch you want to delete: 3
Branch\Month:        1       2       3       4       5       6       7       8       9      10      11      12
Branch 1:        275.0   134.0   475.0   396.0   246.0    77.0    63.5    98.0   365.0   446.0   491.0   431.0
Branch 2:        218.0    40.0   199.0   279.0   325.0   175.0   407.0   427.0   471.0   125.0   334.0   464.0
Branch 3:        999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0
**------------------------------------**
**Please pick of the following options**
**------------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice:
```

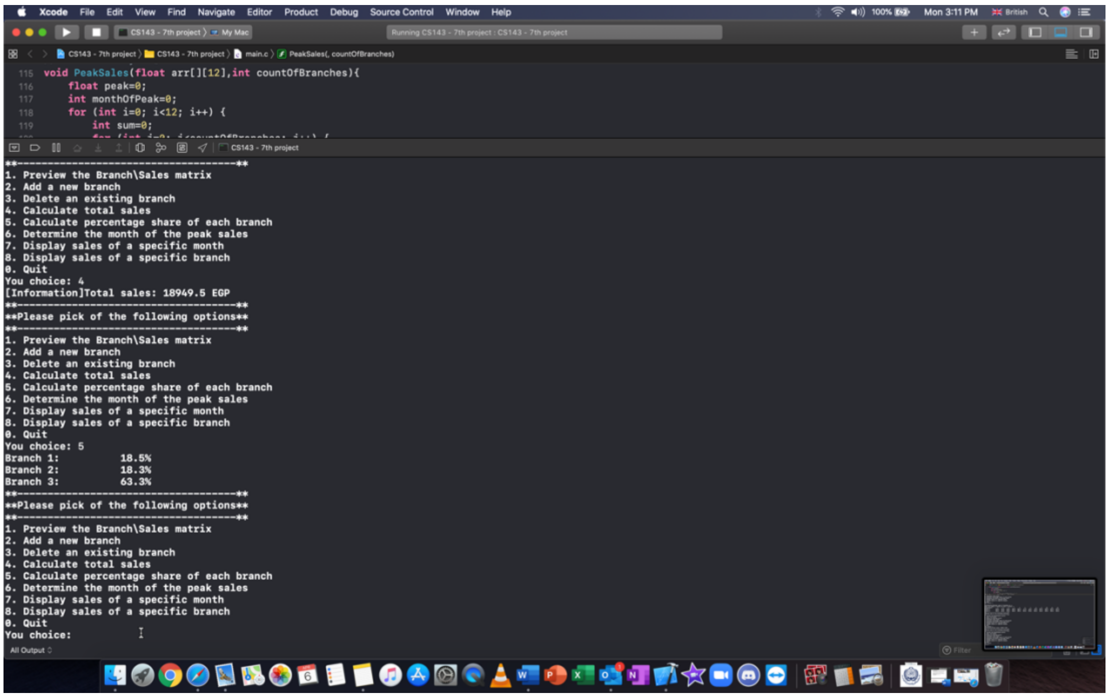
4.6 Calculating total sales



```c
void PeakSales(float arr[][12],int countOfBranches){
    float peak=0;
    int monthOfPeak=0;
    for (int i=0; i<12; i++) {
        int sum=0;
```

```
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: 3

[Information!]Current count of branches is: 4
Input the number of Branch you want to delete: 3
Branch\Month:        1       2       3       4       5       6       7       8       9      10      11      12
Branch 1:        275.0   134.0   475.0   396.0   246.0    77.0    63.5    98.0   365.0   446.0   491.0   431.0
Branch 2:        218.0    40.0   199.0   279.0   325.0   175.0   407.0   427.0   471.0   125.0   334.0   464.0
Branch 3:        999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0   999.0
**------------------------------------**
**Please pick of the following options**
**------------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: 4
[Information]Total sales: 18949.5 EGP
**------------------------------------**
**Please pick of the following options**
**------------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice:
```

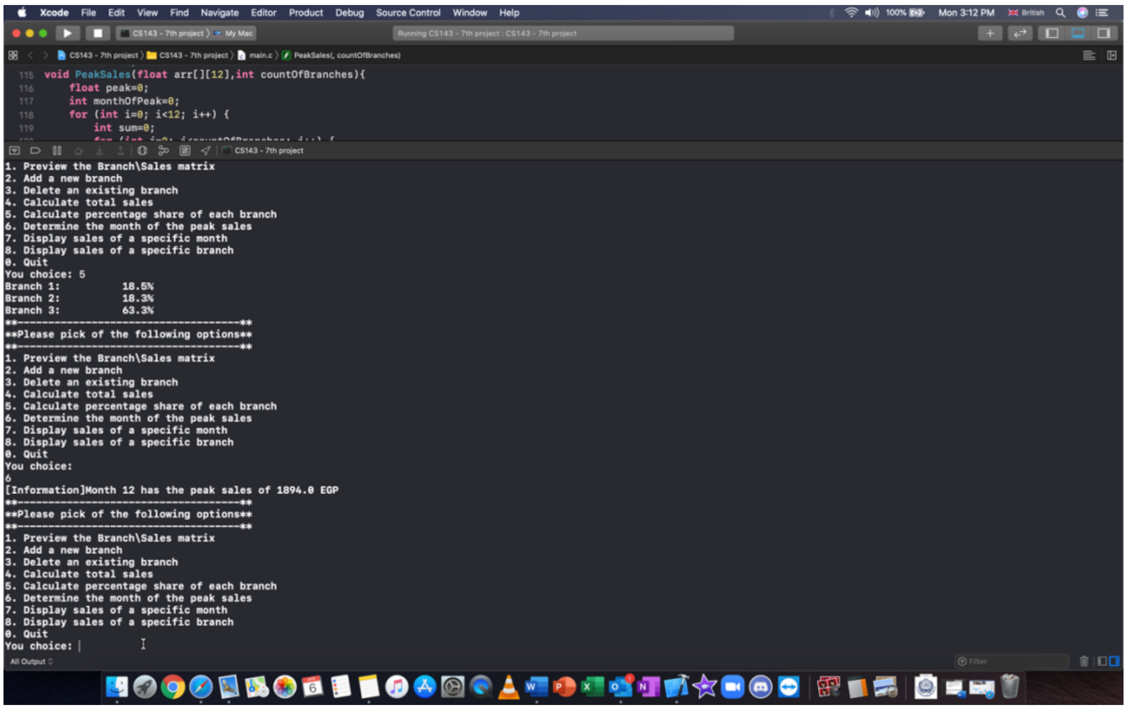
4.7 Sales percentage function



```c
void PeakSales(float arr[][12],int countOfBranches){
    float peak=0;
    int monthOfPeak=0;
    for (int i=0; i<12; i++) {
        int sum=0;
```

```
**----------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: 4
[Information]Total sales: 18949.5 EGP
**----------------------------------**
**Please pick of the following options**
**----------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: 5
Branch 1:          18.5%
Branch 2:          18.3%
Branch 3:          63.3%
**----------------------------------**
**Please pick of the following options**
**----------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice:
```

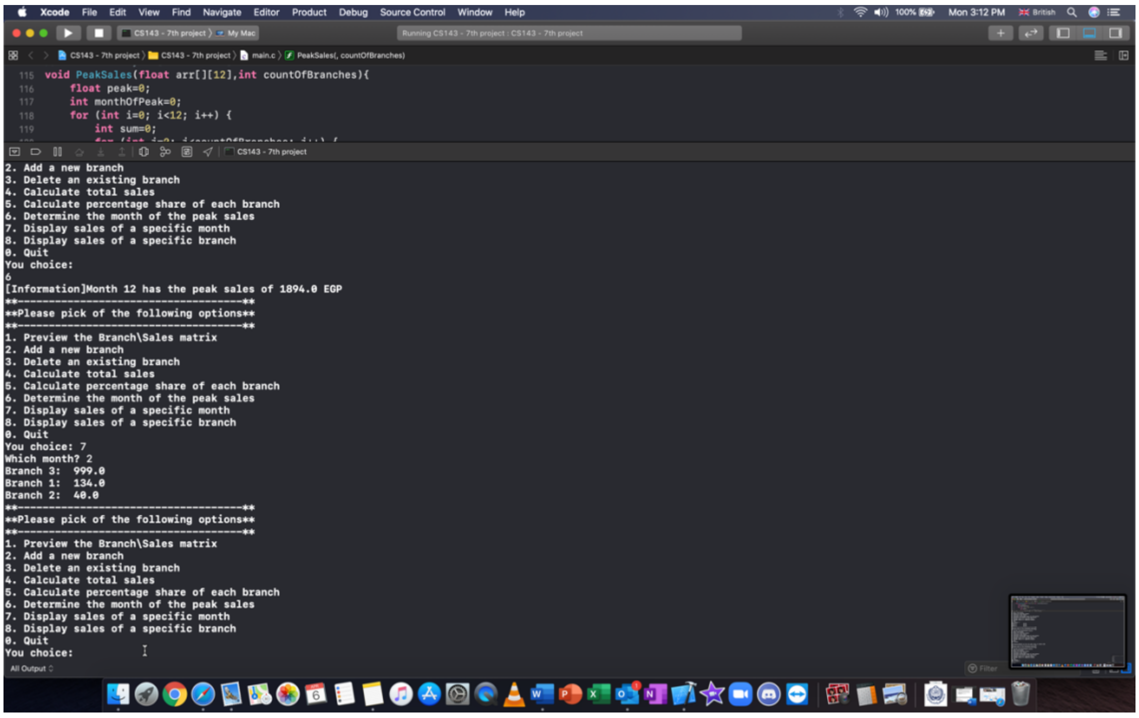
4.8 Peak sales function



```c
void PeakSales(float arr[][12],int countOfBranches){
    float peak=0;
    int monthOfPeak=0;
    for (int i=0; i<12; i++) {
        int sum=0;
```

```
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: 5
Branch 1:          18.5%
Branch 2:          18.3%
Branch 3:          63.3%
**----------------------------------**
**Please pick of the following options**
**----------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice:
6
[Information]Month 12 has the peak sales of 1894.0 EGP
**----------------------------------**
**Please pick of the following options**
**----------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: |
```

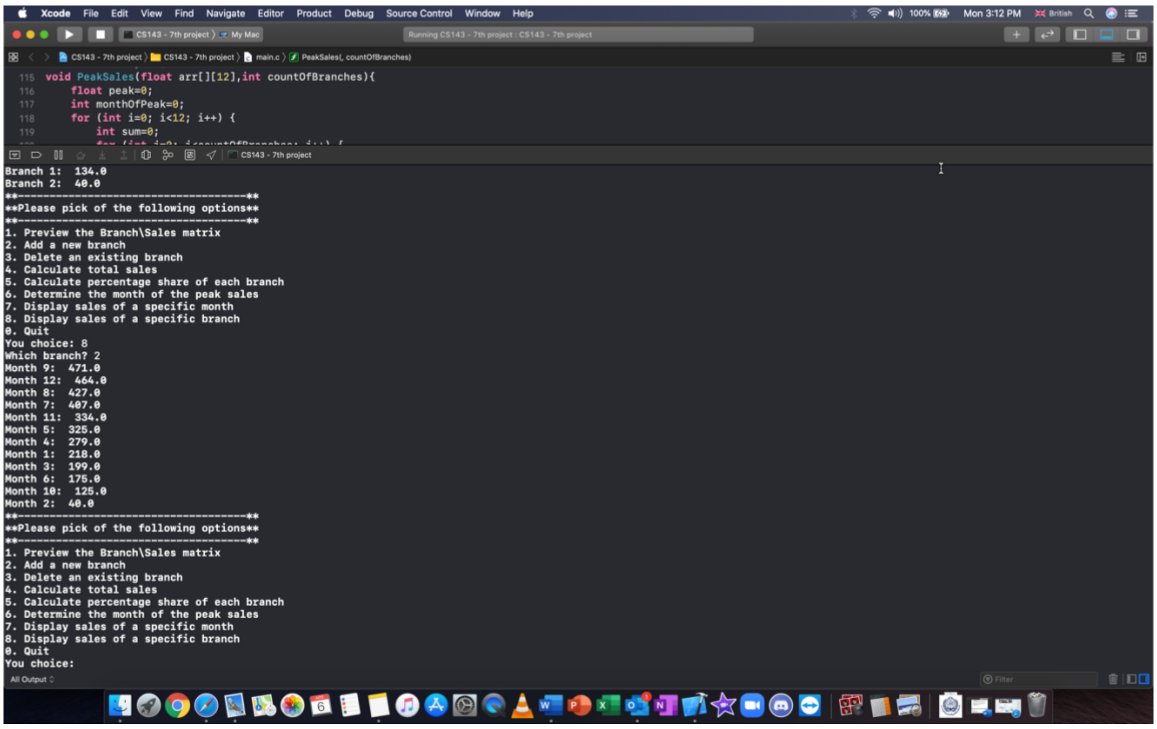
4.9 Sales of specific month



```
void PeakSales(float arr[][12],int countOfBranches){
    float peak=0;
    int monthOfPeak=0;
    for (int i=0; i<12; i++) {
        int sum=0;
```

```
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice:
6
[Information]Month 12 has the peak sales of 1894.0 EGP
**-----------------------------------**
**Please pick of the following options**
**-----------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: 7
Which month? 2
Branch 3:  999.0
Branch 1:  134.0
Branch 2:  40.0
**-----------------------------------**
**Please pick of the following options**
**-----------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice:
```

4.10 Sales of specific branch



```
void PeakSales(float arr[][12],int countOfBranches){
    float peak=0;
    int monthOfPeak=0;
    for (int i=0; i<12; i++) {
        int sum=0;
```

```
Branch 1:  134.0
Branch 2:  40.0
**-----------------------------------**
**Please pick of the following options**
**-----------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice: 8
Which branch? 2
Month 9:  471.0
Month 12:  464.0
Month 8:  427.0
Month 7:  407.0
Month 11:  334.0
Month 5:  325.0
Month 4:  279.0
Month 1:  218.0
Month 3:  199.0
Month 6:  175.0
Month 10:  125.0
Month 2:  40.0
**-----------------------------------**
**Please pick of the following options**
**-----------------------------------**
1. Preview the Branch\Sales matrix
2. Add a new branch
3. Delete an existing branch
4. Calculate total sales
5. Calculate percentage share of each branch
6. Determine the month of the peak sales
7. Display sales of a specific month
8. Display sales of a specific branch
0. Quit
You choice:
```

Thank you