DATA STRUCTURE AND ALGORITHM REVIEW

VELLORE INSTITUTE OF TECHNOLOGY, VELLORE

PIZZA DELIVERY MANAGEMENT SYSTEM IN VIT UNIVERSITY

SUBMITTED BY:-

P.RADHA KRISHNA KASYAP

Reg.no:18BCE0016

Submitted to:

Govinda.K

ABSTRACT:-

In this project, we aim to develop and implement a Pizza delivery in VIT management system using linear data structures arrays etc. An Array, is a data structure consisting of a collection of elements (values or variables), each identified by at least one array index or key. An array is stored such that the position of each element can be computed from its index tuple by a mathematical formula.

Structure is collection of different data type. An object of structure represents a single record in memory, if we want more than one record of structure type, we have to create an array of structure or object. As we know, an array is a collection of similar type, therefore an array can be of structure type.

An Array of Structures in C can be defined as the collection of multiple structures variables where each variable contains information about different entities. The array of structures in C are used to store information about multiple entities of different data types. The array of structures is also known as the collection of structures.

Literature review:

As"18century century hef, assumed by generations of historians to be the world%s first restaurateur, is a figment of the &rench imagination, according to an American academic who spent four years trying to track him down. In a newly published book which has provoked horror in the rarefied world of $arisian gastronomes, (pang, who teaches at university college, london, has challenged &rench historians to produce evidence to back up the claim that a French man named L was responsible for inventing the country best loved social institution. &or more than -NN years /r Loulanger has been credited with opening the world first Paris restaurant in !012.[4] L (pang has been unable to find any evidence that he ever existed. 3This man named Lo simply never appears in any of the sources I have examined,3 she told The Telegraph. 4er investigations are something of an embarrassment for some of &ra most eminent food experts. The commercial activities of the elusive /r Loulanger have been respectfully chronicled in countless histories of &rench cuisine, all of which may now need substantial revision[2].

According to the 5uide 5ourmand de la & France , a standard reference work for French restaurants, /r Loulanger set up his establishment in what is now the in Paris first arrondissement. Above the front door he is reported to have placed a sign stating6 3Loulanger d7bite des restaurants divins3 83Loulanger provides divine sustenance39, so becoming the first businessman to use the word 3restaurant3 8albeit in its original meaning9 to describe a place where food can be had as well as the first to offer a choice of dishes to customers.[3] n a white sauce3 and attributed some of the success of the new restaurant to his beautiful wife, who allegedly attracted the admiring attentions of the writer[1]

[1]
https://www.researchgate.net/publication/228254814_Hotel_Revenue_Management_-_A_Critical_Literature_Review

[2]

http://shodhganga.inflibnet.ac.in/bitstream/10603/149056/8/08_chap

ter%202.pdf

[3]
http://shodhganga.inflibnet.ac.in/bitstream/10603/184744/13/13_cha
pter-2.pdf

<u>AIM:-</u>

To maximize profits by providing a better and faster delivery.

<u>OBJECTIVE:-</u>

To make the use of latest algorithms studied to increase the efficiency of the pizza delivery system.

<u>APPLICABILITY:-</u>

In today's digital world of technology and competition, there is always a room for innovation and improvement.

This project is aimed at providing a better and faster service to the customers in catering business.

This system can be used to provide a better delivery in VIT university for various business like local pizza outlet, grocery delivery and also for other delivery services like amazon, flipkart etc.

INTRODUCTION:-

Nowadays, digital business platforms are very popular and save us much effort and time in our daily life. E-commerce companies such as Amazon and eBay could deliver goods to customers very efficiently. On the one hand, customers could select goods and place orders online without visiting the shop, which is usually time-consuming. Besides, they do not need to carry the goods to home. Instead, the shop would deliver the goods and save customers' efforts. On the other hand, using digital business platforms could make it more convenient for shop owners to manage orders, collect and analyze data and provide better service. In the catering industry, the demand of combining the convenience of digital business with their traditional delivery service is increasingly growing. Unlike common e-commerce companies, the restaurants usually could deliver food in less than half an hour and actually saves customers' time when compared to visiting the restaurants.

In order to stand out in the digital business trend of catering industry and provide more satisfying service, we designed this **Pizza Delivery in Vit Management System**. Customers usually expect fast delivery and food in good condition to eat. So in order to adapt to customers' expectation and earn more profits for restaurant owners, we improved the traditional digital business platform , by designing a new and improvised delivery system through which restaurants can make more money by providing good service, which also helps them stands out among competitors. Delivery staffs will be provided information for more efficient operations.

EXISTING DRAWBACKS

1.Conventional delivery systems mainly focus on first come first serve basis for delivery of orders to different locations .

2.This makes it less economical for the local pizzeria as the deliveryman doesn’t follow a predetermined path leading to increase in costs.

PROPOSED METHOD AND ADVANTAGES

1.Our method sorts orders according to time taken to reach the destination and provides a path for the deliveryman to follow to deliver orders, so that he can execute his task with least effort and time with maximum efficiency.

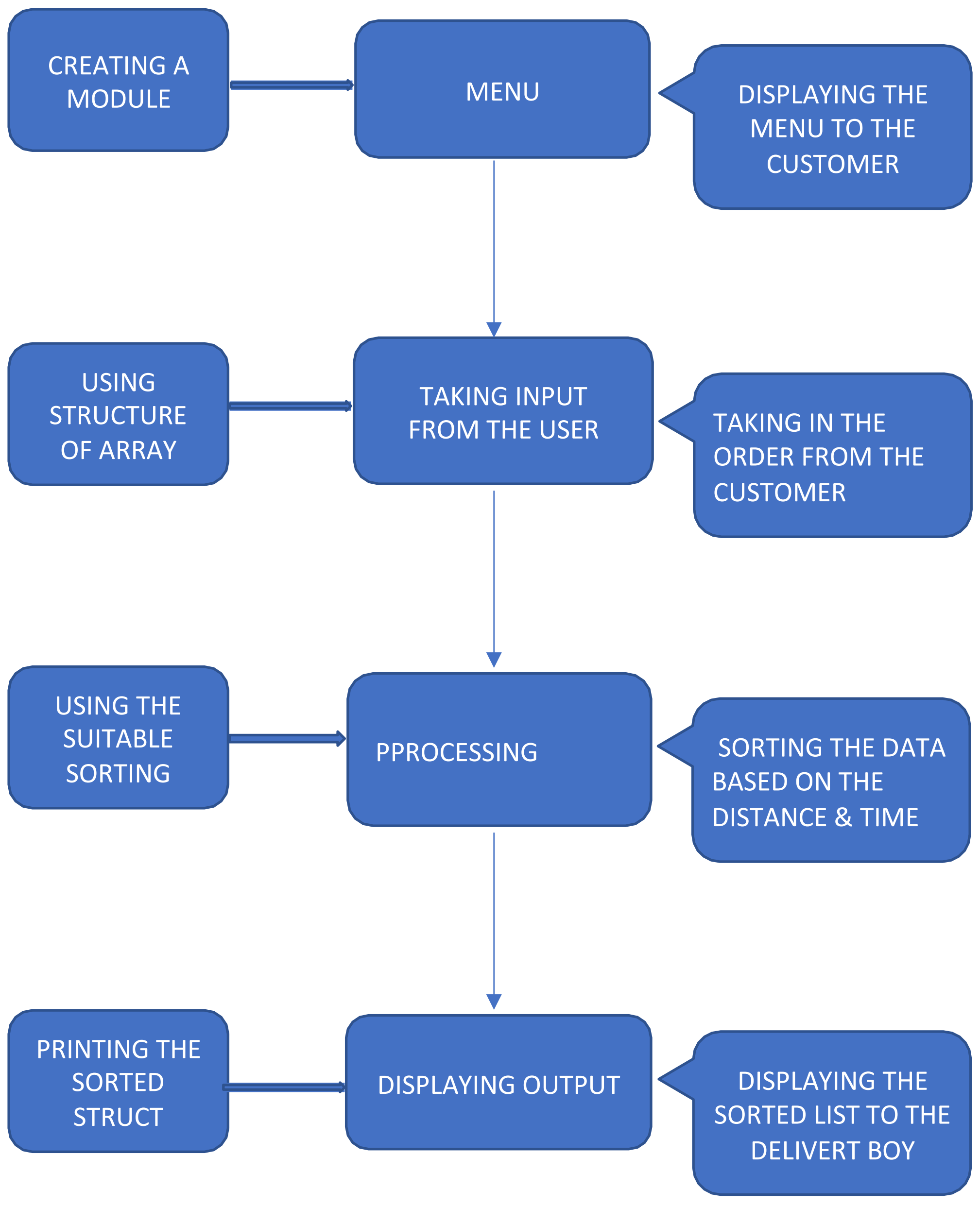
2.For the pizza as a whole this translates to maximum number of orders being delivered in minimum time.

3.The main advantage is that the local pizza incurs less cost for delivery task.

4.Customers usually expect fast delivery and food in good condition to eat.

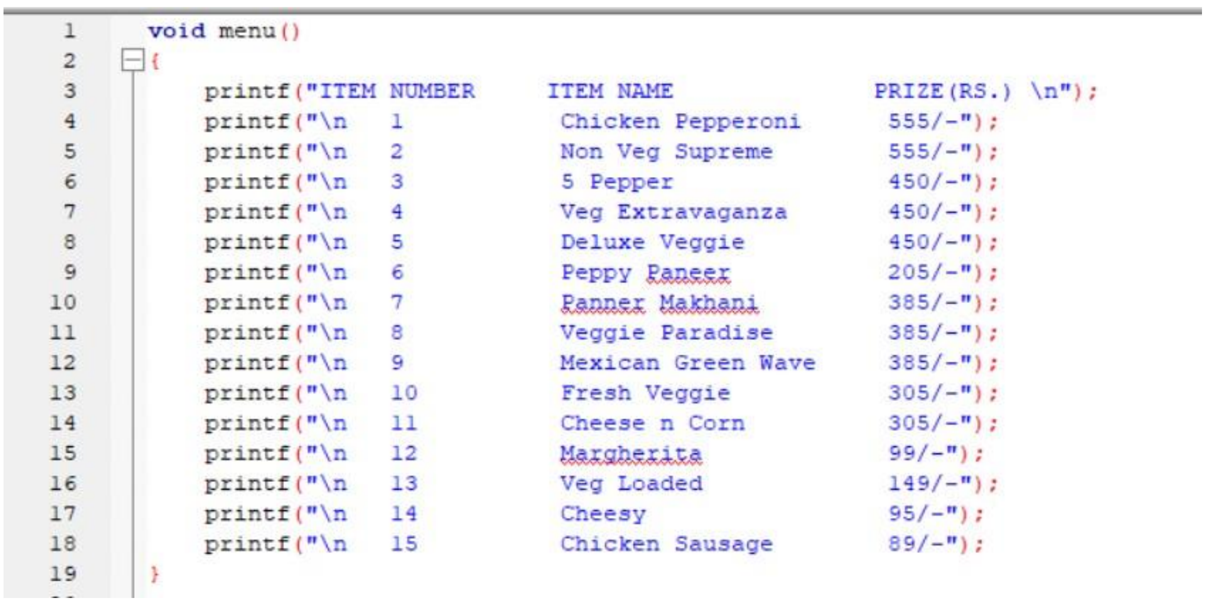
5.This application thus helps to adapt to customers’ expectation and earn more profits for restaurant owners.

PROPOSED SYSTEM ARCHITECTURE


CREATING A MODULE
MENU
DISPLAYING THE MENU TO THE CUSTOMER
USING STRUCTURE OF ARRAY
TAKING INPUT FROM THE USER
TAKING IN THE ORDER FROM THE CUSTOMER
USING THE SUITABLE SORTING
PPROCESSING
SORTING THE DATA BASED ON THE DISTANCE & TIME
PRINTING THE SORTED STRUCT
DISPLAYING OUTPUT
DISPLAYING THE SORTED LIST TO THE DELIVERT BOY

MODULE USED

No predefined module is used in the application but a module was created to store the menu of the pizzeria.



```
void menu()
{
    printf("ITEM NUMBER     ITEM NAME                PRIZE(RS.)  \n");
    printf("\n   1            Chicken Pepperoni         555/-");
    printf("\n   2            Non Veg Supreme           555/-");
    printf("\n   3            5 Pepper                  450/-");
    printf("\n   4            Veg Extravaganza          450/-");
    printf("\n   5            Deluxe Veggie             450/-");
    printf("\n   6            Peppy Paneer              205/-");
    printf("\n   7            Panner Makhani            385/-");
    printf("\n   8            Veggie Paradise           385/-");
    printf("\n   9            Mexican Green Wave        385/-");
    printf("\n   10           Fresh Veggie              305/-");
    printf("\n   11           Cheese n Corn             305/-");
    printf("\n   12           Margherita                99/-");
    printf("\n   13           Veg Loaded                149/-");
    printf("\n   14           Cheesy                    95/-");
    printf("\n   15           Chicken Sausage           89/-");
}
```

1) ALGORITHM :

1)First we have to create a module to store food menu.

2)We have to create structure to store the details of customers and their order details.

```
struct customer
{
   char name[20];
   double mobile;
   char block;
   int order;
   int cost;
   int d;
};
```

3) By using switch cases we will come to know the amount the customer should

pay.

```
int cost_pizza (int num)
{
   int cost;
   switch(num)
   {
         Case 1:
         Cost= x;
         break;
}
return(cost);
```

4) We use clock_t start_time = clock(); to get the present time.
   And we take int msec= 60*1500; so at any cost we will get the output after 1.5 Minuites.

5) By using this while(i<s && clock()< start_time+msec) condition we will take Inputs from the customer.

6) By using else conditions we are assigning integer values to every block.
   Like

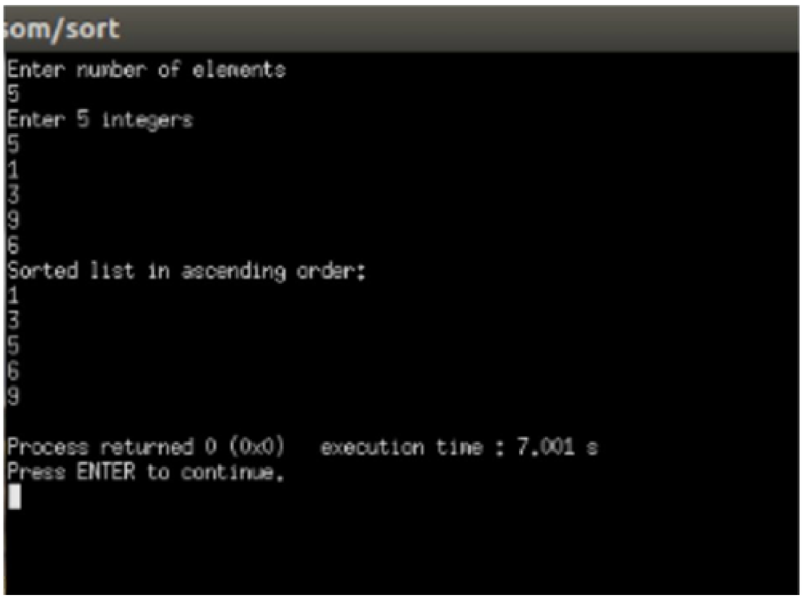
```
         if(arr[i].block=='A' || arr[i].block=='a')
                          {
                                  arr[i].d=7;
                          }
```

7) By using insertion sort:

Insertion sort algorithm somewhat resembles selection sort. Array is imaginary divided into two parts-sorted one and unsorted one. At the beginning, sorted part contains first element of the array and unsorted one contains the rest. At every step, algorithm takes first element in the unsorted part and inserts it to the right place of the sorted one. When unsorted part becomes empty, algorithm stops.

```
for (i = 1; i < p; i++) {
        key = arr[i];
        j = i - 1;

        /* Move elements of arr[0..i-1], that are
          greater than key, to one position ahead
          of their current position */
        while (j >= 0 && arr[j].d > key.d) {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
```

<u>REALTIME EXECUTION OFTHE INSERTION SORT</u>:



```
om/sort
Enter number of elements
5
Enter 5 integers
5
1
3
9
6
Sorted list in ascending order:
1
3
5
6
9

Process returned 0 (0x0)   execution time : 7.001 s
Press ENTER to continue.
```

8) By using this display function:

void display(struct customer arr[num], int p)

We will display the shortest path to travel by the pizza delivery boy and the total amount the customer have to pay.

2)Algorithm used :

1)First we have to create a module to store food menu.

2)We have to create structure to store the details of customers and their order details.

```
struct customer
{
   char name[20];
   double mobile;
   char block;
   int order;
   int cost;
   int d;
};
```

3) By using switch cases we will come to know the amount the customer should pay.

```
int cost_pizza (int num)
{
   int cost;
   switch(num)
   {
        Case 1:
        Cost= x;
        break;
}
return(cost);
```

4) We use clock_t start_time = clock(); to get the present time.
   And we take int msec= 60*1500; so at any cost we will get the output after 1.5 Minuites.

5) By using this while(i<s && clock()< start_time+msec) condition we will take Inputs from the customer.

6) By using else conditions we are assigning integer values to every block.
   Like

```
        if(arr[i].block=='A' || arr[i].block=='a')
                      {
                              arr[i].d=7;
                      }
```

7) By using bubblesort algorithm:

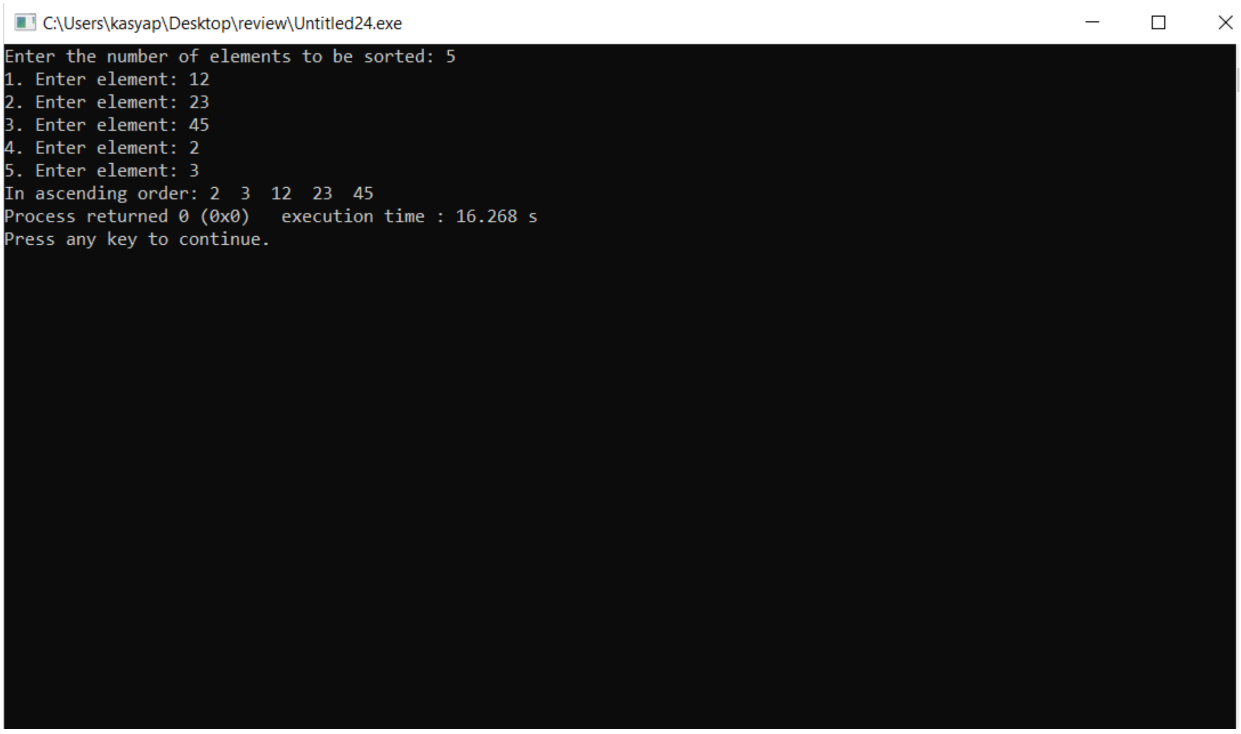
Bubble sort algorithm starts by comparing the first two elements of an array and swapping if necessary, i.e., if you want to sort the elements of array in ascending order and if the first element is greater than second then, you need to swap the elements but, if the first element is smaller than second, you mustn't swap the element. Then, again second and third elements are compared and swapped if it is necessary and this process go on until last and second last element is compared and swapped. This completes the first step of bubble sort.

If there are n elements to be sorted then, the process mentioned above should be repeated n-1 times to get required result. But, for better performance, in second step, last and second last elements are not compared becuase, the proper element is automatically placed at last after first step. Similarly, in third step, last and second last and second last and third last elements are not compared and so on.

A figure is worth a thousand words so, acknowledge this figure for better understanding of bubble sort.

```
for (i = 0; i < p - 1; i++)
   {
      for (j = i+1; j <p; j++)
      {
         if (arr[i].d > arr[j].d)
         {
            temp = arr[j];
            arr[j] = arr[i];
            arr[i] = temp;
         }
      }
   }
```

<u>REALTIME EXECUTION OF THE BUBBLESORT</u>:



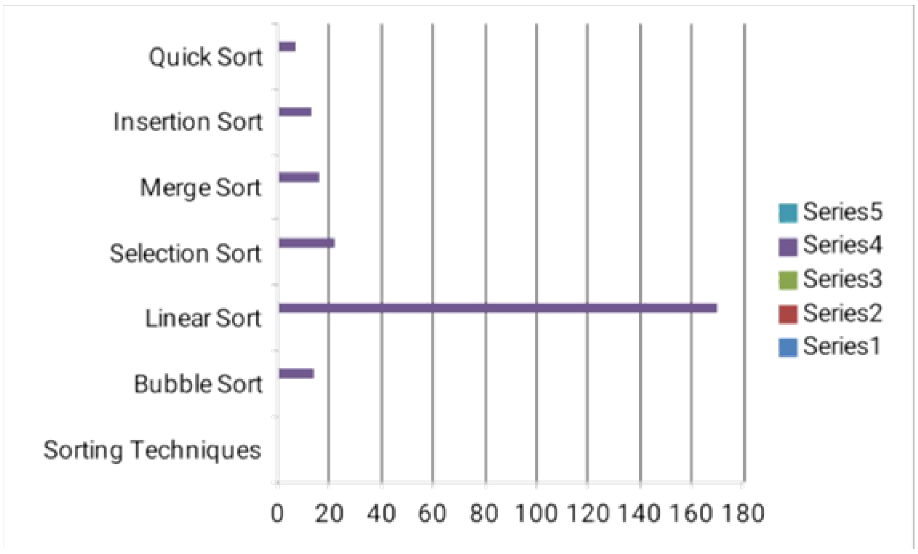
```
Enter the number of elements to be sorted: 5
1. Enter element: 12
2. Enter element: 23
3. Enter element: 45
4. Enter element: 2
5. Enter element: 3
In ascending order: 2  3  12  23  45
Process returned 0 (0x0)   execution time : 16.268 s
Press any key to continue.
```

8) By using this display function:

void display(struct customer arr[num], int p)

We will display the shortest path to travel by the pizza delivery boy and the total amount the customer have to pay.

<u>BASED ON THE EXECUTION TIME OF ALL THE SORTING TECHNIQUES, HERE IS A GRAPH TO GIVE THE BEST TECHNIQUE:</u>



CODE 1 : BY USING INSERTION SORT:Tomasulo algorithm

```
#include <stdio.h>
#include <stdlib.h>
#include<string.h>
#include <time.h>


void menu()
{
   printf("ITEM NUMBER     ITEM NAME             PRIZE(RS.) \n");
   printf("\n   1         Chicken Pepperoni      555/-");
   printf("\n   2         Non Veg Supreme        555/-");
   printf("\n   3         5 Pepper             450/-");
   printf("\n   4         Veg Extravaganza       450/-");
   printf("\n   5         Deluxe Veggie         450/-");
   printf("\n   6         Peppy Paneer          205/-");
   printf("\n   7         Panner Makhani        385/-");
   printf("\n   8         Veggie Paradise       385/-");
   printf("\n   9         Mexican Green Wave     385/-");
   printf("\n   10         Fresh Veggie          305/-");
   printf("\n   11         Cheese n Corn          305/-");
   printf("\n   12         Margherita            99/-");
   printf("\n   13         Veg Loaded            149/-");
   printf("\n   14         Cheesy               95/-");
   printf("\n   15         Chicken Sausage        89/-");
   printf("\n   16         Chicken Paradise       99/-");
   printf("\n   17         Chicken Chef Special    89/-");
   printf("\n   18         Veg Farmhouse         159/-");
   printf("\n   19         Anish Special         189/-");
```

```
    printf("\n   20        Chicken Cheese        289/-");
    printf("\n   21        Atharva Chicken        99/-");
    printf("\n   22        Duck Chicken Sausage   259/-");
    printf("\n   23        Duck Paradise         149/-");
    printf("\n   24        Cheese blast          99/-");
    printf("\n   25        Chicken CheeseSausage  159/-");
}


struct customer
{
    char name[20];
    double mobile;
    char block;
    int order;
    int cost;
    int d;
};
int num =0;
int cost_pizza (int num)
{
    int cost;
    switch(num)
    {
    case 1:
        cost= 555;
        break;
    case 2:
        cost= 555;
        break;
    case 3:
        cost= 450;
        break;
    case 4:
        cost= 450;
        break;
    case 5:
        cost= 450;
        break;
    case 6:
        cost= 205;
        break;
    case 7:
        cost= 385;
        break;
    case 8:
        cost= 385;
        break;
    case 9:
        cost= 385;
        break;
    case 10:
        cost= 305;
        break;
    case 11:
        cost= 305;
        break;
    case 12:
        cost= 99;
        break;
    case 13:
        cost= 149;
        break;
    case 14:
        cost= 95;
        break;
    case 15:
        cost= 89;
        break;
    case 16:
        cost= 99;
        break;
    case 17:
        cost= 89;
        break;
    case 18:
            cost= 159;
            break;
        case 19:
            cost=189;
            break;
        case 20:
            cost= 289;
            break;
        case 21:
            cost= 99;
            break;
        case 22:
            cost= 259;
            break;
        case 23:
            cost= 149;
            break;
        case 24:
            cost= 99;
            break;
        case 25:
            cost= 159;
            break;
        };
        return(cost);
    }
    int accept(struct customer arr[8] , int s)
    {
        int msec= 60*1500;
        clock_t start_time = clock();
        int i=0;
        while(i<s && clock()< start_time+msec)
        {
             printf("\nEnter your name ");
            scanf("%s",&arr[i].name);
              printf("Enter your contact number ");
                scanf("%lf",&arr[i].mobile);
                printf("Enter your block ");
```

```
scanf(" %c",&arr[i].block);
//arr[i].block=strupr(arr[i].block);
 printf("Enter your order item number ");
    scanf("%d",&arr[i].order);
     if(arr[i].block=='A' || arr[i].block=='a')
                {
                        arr[i].d=7;
                }

                else if(arr[i].block=='L' || arr[i].block=='l')
                {
                        arr[i].d=15;
                }
                else if(arr[i].block=='M' || arr[i].block=='m')
                {
                        arr[i].d=16;
                }
                else if(arr[i].block=='N' || arr[i].block=='n')
                {
                        arr[i].d=18;
                }
                else if(arr[i].block=='B'|| arr[i].block=='b')
                {
                        arr[i].d=8;
                }
                else if(arr[i].block=='D' || arr[i].block=='d')
                {
                        arr[i].d=10;
                }
                else if(arr[i].block=='E' || arr[i].block=='e')
                {
                        arr[i].d=11;
                }
                else if(arr[i].block=='G' || arr[i].block=='g')
                {
                        arr[i].d=13;
                }
                else if(arr[i].block=='H' || arr[i].block=='h')
```

```
                        {
                                arr[i].d=5;
                        }
                        else if(arr[i].block=='F' || arr[i].block=='f')
                        {
                                arr[i].d=9;
                        }

   i++;

   }
   num =i;
   return(num);
}

void insertionSort(struct customer arr[num], int p)
{
   int i, j;
   struct customer key;

   for (i = 1; i < p; i++) {
      key = arr[i];
      j = i - 1;

      /* Move elements of arr[0..i-1], that are
        greater than key, to one position ahead
        of their current position */
      while (j >= 0 && arr[j].d > key.d) {
         arr[j + 1] = arr[j];
         j = j - 1;
      }
      arr[j + 1] = key;
   }
}
void display(struct customer arr[num], int p)
{
   int i;
   printf("\nOrder Number\tCustomer
name\tMobile_Number\tBlock\tOrder_Number\tCost\n");
   for (i =0; i<p; i++)
   {
      arr[i].cost=cost_pizza(arr[i].order);
      printf("%d\t\t%s\t\t%.0lf\t%c\t\t%d\t%d\n",i+1, arr[i].name,
arr[i].mobile, arr[i].block,arr[i].order,arr[i].cost);
   }
}

int main()
{
   int s=8;
   int p;
   struct customer arr[8];
   int accept(struct customer arr[], int s);
  // void bsortDesc(struct customer arr[],int s);
   void display(struct customer arr[],int s);
   menu();
   p = accept(arr,s);
   getch();
   system("cls");
   insertionSort(arr,p);
   display(arr,p);
   getch();

   return 0;
}
```

<u>CODE 2 : BY USING BUBBLESORT:</u>

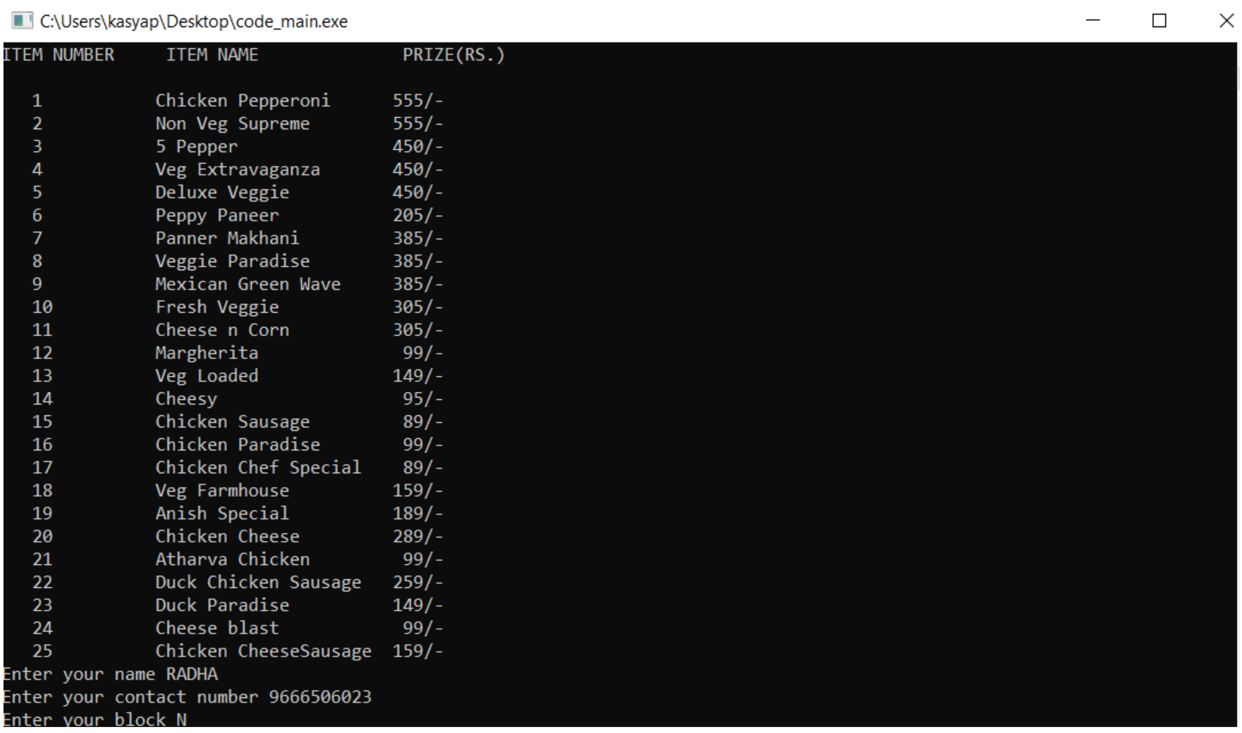
```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void menu()
{
   printf("ITEM NUMBER     ITEM NAME             PRIZE(RS.) \n");
   printf("\n   1          Chicken Pepperoni      555/-");
   printf("\n   2          Non Veg Supreme        555/-");
   printf("\n   3         5 Pepper            450/-");
   printf("\n   4         Veg Extravaganza      450/-");
   printf("\n   5         Deluxe Veggie        450/-");
   printf("\n   6         Peppy Paneer         205/-");
   printf("\n   7         Panner Makhani        385/-");
   printf("\n   8         Veggie Paradise       385/-");
   printf("\n   9         Mexican Green Wave    385/-");
   printf("\n   10         Fresh Veggie         305/-");
   printf("\n   11         Cheese n Corn         305/-");
   printf("\n   12         Margherita            99/-");
   printf("\n   13         Veg Loaded           149/-");
   printf("\n   14         Cheesy              95/-");
   printf("\n   15         Chicken Sausage        89/-");
   printf("\n   16         Chicken Paradise       99/-");
   printf("\n   17         Chicken Chef Special    89/-");
   printf("\n   18         Veg Farmhouse         159/-");
   printf("\n   19         Anish Special        189/-");
   printf("\n   20         Chicken Cheese        289/-");
   printf("\n   21         Atharva Chicken        99/-");
   printf("\n   22         Duck Chicken Sausage   259/-");
   printf("\n   23         Duck Paradise         149/-");
   printf("\n   24         Cheese blast          99/-");
   printf("\n   25         Chicken CheeseSausage  159/-");
}
struct customer
{
   char name[20];
   double mobile;
   char block;
   int order;
   int cost;
   int d;
};
int num =0;
int cost_pizza (int n)
{
   int cost;
   switch(n)
```

```
{
case 1:
   cost= 555;
   break;
case 2:
   cost= 555;
   break;
case 3:
   cost= 450;
   break;
case 4:
   cost= 450;
   break;
case 5:
   cost= 450;
   break;
case 6:
   cost= 205;
   break;
case 7:
   cost= 385;
   break;
case 8:
   cost= 385;
   break;
case 9:
   cost= 385;
   break;
case 10:
   cost= 305;
   break;
case 11:
   cost= 305;
   break;
case 12:
   cost= 99;
   break;
case 13:
   cost= 149;
   break;
case 14:
   cost= 95;
   break;
case 15:
   cost= 89;
   break;
case 16:
   cost= 99;
   break;
case 17:
   cost= 89;
   break;
case 18:
   cost= 159;
   break;
case 19:
   cost=189;
   break;
case 20:
   cost= 289;
   break;
case 21:
   cost= 99;
   break;
case 22:
   cost= 259;
   break;
case 23:
   cost= 149;
   break;
case 24:
   cost= 99;
   break;
case 25:
   cost= 159;
   break;
```

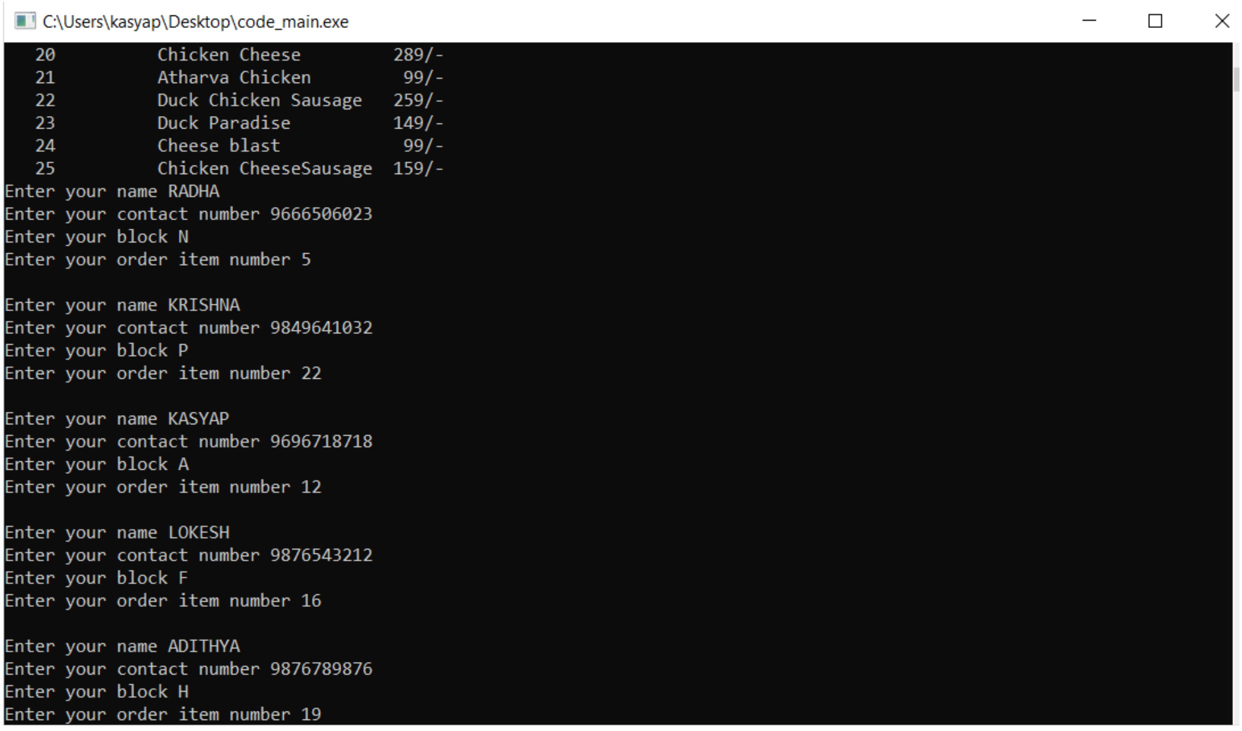
```
   };
   return(cost);
}
int accept(struct customer arr[8] , int s)
{
   int msec= 60*1500;
   clock_t start_time = clock();
   int i=0;
   while(i<s && clock()< start_time+msec)
   {
       printf("\nEnter your name ");
      scanf("%s",&arr[i].name);
        printf("Enter your contact number ");
         scanf("%lf",&arr[i].mobile);
         printf("Enter your block ");
         scanf(" %c",&arr[i].block);
         //arr[i].block=strupr(arr[i].block);
          printf("Enter your order item number ");
            scanf("%d",&arr[i].order);
             if(arr[i].block=='A' || arr[i].block=='a')
                        {
                                arr[i].d=7;
                        }

                        else if(arr[i].block=='L' || arr[i].block=='l')
                        {
                                arr[i].d=15;
                        }
                        else if(arr[i].block=='k' || arr[i].block=='K')
                        {
                                arr[i].d=14;
                        }
                        else if(arr[i].block=='M' || arr[i].block=='m')
                        {
                                arr[i].d=16;
                        }
                        else if
(arr[i].block=='p'||arr[i].block=='P'||arr[i].block=='q'||arr[i].block=='
Q')
        {
          arr[i].d=17;
        }
                        else if(arr[i].block=='N' || arr[i].block=='n')
                        {
                                arr[i].d=18;
                        }
                        else if(arr[i].block=='B'|| arr[i].block=='b')
                        {
                                arr[i].d=8;
                        }
                        else if(arr[i].block=='D' || arr[i].block=='d')
                        {
                                arr[i].d=10;
                        }
                        else if(arr[i].block=='E' || arr[i].block=='e')
                        {
                                arr[i].d=11;
                        }
                        else if(arr[i].block=='G' || arr[i].block=='g')
                        {
                                arr[i].d=13;
                        }
                        else if(arr[i].block=='H' || arr[i].block=='h')
                        {
                                arr[i].d=5;
                        }
                        else if(arr[i].block=='F' || arr[i].block=='f')
                        {
                                arr[i].d=9;
                        }
                        else if(arr[i].block=='j'|| arr[i].block=='J')
        {
          arr[i].d=4;
        }

  i++;
```

```
   }
   num =i;
   return(num);
}





void bsortDesc(struct customer arr[num],int p)
{
   int i, j;
   struct customer temp;

   for (i = 0; i < p - 1; i++)
   {
      for (j = i+1; j <p; j++)
      {
         if (arr[i].d > arr[j].d)
         {
            temp = arr[j];
            arr[j] = arr[i];
            arr[i] = temp;
         }
      }
   }
}

void display(struct customer arr[num], int p)
{
   int i;
   printf("\nOrder Number\tCustomer
name\tMobile_Number\tBlock\tOrder_Number\tCost\n");
   for (i =0; i<p; i++)
   {
      arr[i].cost=cost_pizza(arr[i].order);
      printf("%d\t\t%s\t\t%.0lf\t%c\t\t%d\t%d\n",i+1, arr[i].name,
arr[i].mobile, arr[i].block,arr[i].order,arr[i].cost);
   }
}

int main()
{
   int s=8;
   int p;
   struct customer arr[8];
   int accept(struct customer arr[], int s);
   void bsortDesc(struct customer arr[],int s);
   void display(struct customer arr[],int s);
   menu();
   p = accept(arr,s);
   getch();
   system("cls");
   bsortDesc(arr,p);
   display(arr,p);
   getch();

   return 0;
}
```

INPUT:

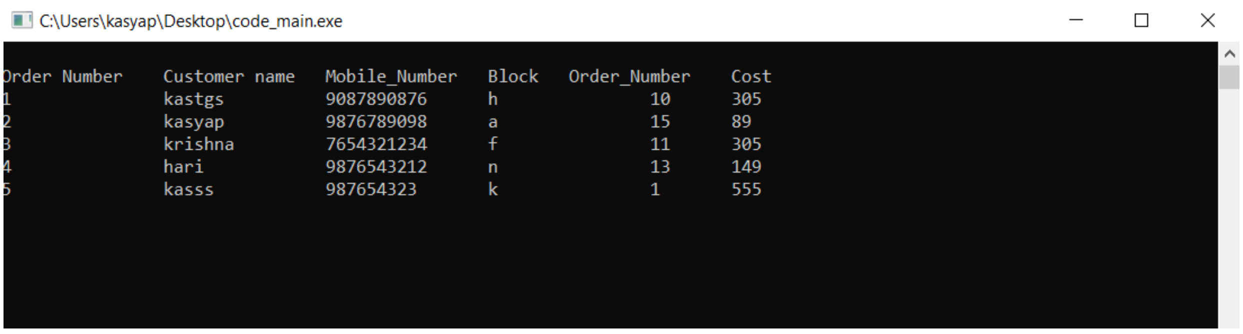

C:\Users\kasyap\Desktop\code_main.exe

```
ITEM NUMBER     ITEM NAME              PRIZE(RS.)

   1           Chicken Pepperoni      555/-
   2           Non Veg Supreme        555/-
   3           5 Pepper               450/-
   4           Veg Extravaganza       450/-
   5           Deluxe Veggie          450/-
   6           Peppy Paneer           205/-
   7           Panner Makhani         385/-
   8           Veggie Paradise        385/-
   9           Mexican Green Wave     385/-
   10          Fresh Veggie           305/-
   11          Cheese n Corn          305/-
   12          Margherita              99/-
   13          Veg Loaded             149/-
   14          Cheesy                  95/-
   15          Chicken Sausage         89/-
   16          Chicken Paradise        99/-
   17          Chicken Chef Special    89/-
   18          Veg Farmhouse          159/-
   19          Anish Special          189/-
   20          Chicken Cheese         289/-
   21          Atharva Chicken         99/-
   22          Duck Chicken Sausage   259/-
   23          Duck Paradise          149/-
   24          Cheese blast            99/-
   25          Chicken CheeseSausage  159/-
Enter your name RADHA
Enter your contact number 9666506023
Enter your block N
```


C:\Users\kasyap\Desktop\code_main.exe

```
   20          Chicken Cheese         289/-
   21          Atharva Chicken         99/-
   22          Duck Chicken Sausage   259/-
   23          Duck Paradise          149/-
   24          Cheese blast            99/-
   25          Chicken CheeseSausage  159/-
Enter your name RADHA
Enter your contact number 9666506023
Enter your block N
Enter your order item number 5

Enter your name KRISHNA
Enter your contact number 9849641032
Enter your block P
Enter your order item number 22

Enter your name KASYAP
Enter your contact number 9696718718
Enter your block A
Enter your order item number 12

Enter your name LOKESH
Enter your contact number 9876543212
Enter your block F
Enter your order item number 16

Enter your name ADITHYA
Enter your contact number 9876789876
Enter your block H
Enter your order item number 19
```

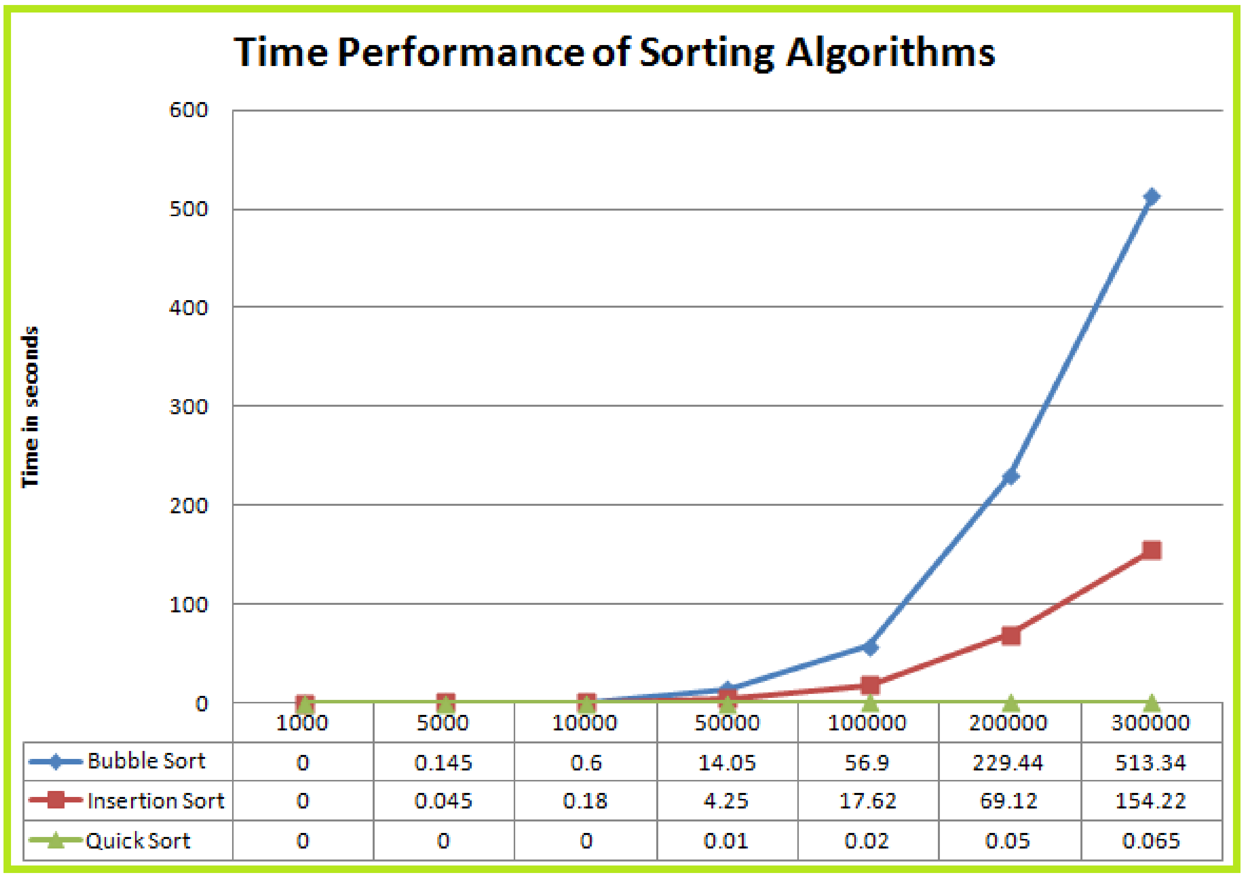
OUTPUT:



```
Order Number     Customer name    Mobile_Number    Block    Order_Number    Cost
1                kastgs           9087890876       h                  10     305
2                kasyap           9876789098       a                  15     89
3                krishna          7654321234       f                  11     305
4                hari             9876543212       n                  13     149
5                kasss            987654323        k                  1      555
```

<u>Comparison between bubble sort and insertion sort algorithm:</u>

<u>Time complexity:</u>



	1000	5000	10000	50000	100000	200000	300000
Bubble Sort	0	0.145	0.6	14.05	56.9	229.44	513.34
Insertion Sort	0	0.045	0.18	4.25	17.62	69.12	154.22
Quick Sort	0	0	0	0.01	0.02	0.05	0.065

CONCLUSION

The delivery management system helps in fulfilling different objectives. From customers' perspective, they get a hot pizza in a relatively shorter time. From the delivery person's perspective, they will get navigation help from the software to be better able to deliver the orders.
For the pizzeria as whole this translates to better delivery and increasing profits.

In conclusion, Pizza Home Management system would provide a convenient, thoughtful and reasonable way for customers to order pizza, and for deliverymen to deliver orders in less time.

FUTURE ENHANCEMENT

There are many ideas which can be added to improvise the project. These ideas are as follows:-

- Creating a better user interface to take order and display information to the customer.
- Creating a complete electronic system to keep food hot while delivering.
- Creating a mobile application for more convivence.
- Integration of GPS module for showing the navigation to the deliverymen.

REFERENCES

www.geeksforgeeks.com
http://www.tutorialdost.com
http://www.cprogrammingnotes.com