

RESTRAUNT MANAGEMENT PROGRAM

GROUP MEMBERS:

Muhammad Rafay 20K-0180

Anas Ali 20K-0352

1

INTRODUCTION:

Our project name is “Restaurant Management.” The idea of making this management system is to turn a manually order taking system into automated. The restaurant was going through an upgrade and wanted to remove the mistakes which would take place while taking order manually like, taking wrong order, forgetting the table number, quoting the wrong total. Hence to solve the issue, this machine-based (automated) system software has been programmed.

USERS:

* Customers
* Chef/waiter

FEATURES:

* Place orders
* Display complete menu
* Offer random discounts
* Bill generation

PROJECT:

The program basically, takes input the order number (table number)/ and customer name so that the orders of different customer wont be mixed. Then menu is displayed. The menu basically consists of different types of dishes according to time. For example breakfast, lunch and dinner. The user orders the food. After completing the order, total bill is calculated for the ordered food and shown immediately. Then all the attributes are then saved in a file so that further process of serving can be done.

RESULTS:

The results of this project our that the restaurant can easily manage orders now and there are no chances of error left until and unless the user makes some error or systematic error occurs. Also, the restaurant can also manage higher number of order even implement this online to take pre-orders of arriving customers.

CODE:

include irvine32.inc

IncludeLib Irvine32.lib

BUFFER\_SIZE = 1000

.data

bill DWORD 0

billheader byte " \*\*\*------------OUR RESTAURANT BILL-------------\*\*\*",0dh,0ah,0

heading BYTE " \*\*\*------------WELCOME TO OUR RESTAURANT-------------\*\*\*",0dh,0ah,0

menu\_list BYTE " \*\*\*\*\*MENU LIST\*\*\*\*\* ",0dh,0ah

BYTE " ",0dh,0ah

BYTE " ",0dh,0ah

BYTE " \* 1. BREAK FAST \*",0dh,0ah

BYTE " \* 2. LUNCH \*",0dh,0ah

BYTE " \* 3. DINNER \*",0dh,0ah

BYTE " \* 4. DRINKS AND SALADS \*",0dh,0ah

BYTE " \* 5. DESSERTS \*",0dh,0ah

BYTE " \* 6. EXIT \*",0dh,0ah,0

bill\_confirm BYTE " 1. WANT MORE FOOD? ",0dh,0ah

BYTE " 2. CREATE BILL ",0dh,0ah,0

bread\_menu BYTE " \* 1. ROGHNI NAAN = RS 30 \*",0dh,0ah

BYTE " \* 2. CHAPATI = RS 10 \*",0dh,0ah

BYTE " \* 3. Exit \*",0dh,0ah,0

breadprices DWORD 30,10

lunch\_menu BYTE " \* 1. CHICKEN BIRYANI = RS 170 PER PLATE \*",0dh,0ah

BYTE " \* 2. CHICKEN KARAHI = RS 180 PER PLATE \*",0dh,0ah

BYTE " \* 3. CHICKEN KORMA = RS 210 PER PLATE \*",0dh,0ah

BYTE " \* 4. CHICKEN PULAO = RS 140 PER PLATE \*",0dh,0ah

BYTE " \* 5. EXIT \*",0dh,0ah,0

lunchprices DWORD 170,180,210,140

breakfast\_menu BYTE " \* 1. PARATHA = RS 30",0dh,0ah

BYTE " \* 2. CHAI = RS 40",0dh,0ah

BYTE " \* 3. HALWA POORI = RS 35",0dh,0ah

BYTE " \* 4. FRIED EGG = Rs 35",0dh,0ah

BYTE " \* 5. EXIT \*",0dh,0ah,0

breakfastprices DWORD 30,40,35,35

fastfood\_menu BYTE " \* 1. ZINGER BURGER = RS 250 \*",0dh,0ah

BYTE " \* 2. CHICKEN BROAST = Rs 270 \*",0dh,0ah

BYTE " \* 3. CHCIKEN BURGER = Rs 150 \*",0dh,0ah

BYTE " \* 4. CLUB SANDWHICH = Rs 170 \*",0dh,0ah

BYTE " \* 5. EXIT \*",0dh,0ah,0

fastfoodprices DWORD 250,270,150,170

drinks\_menu BYTE " \* 1. PEPSI = RS 70 (500 ml) \*",0dh,0ah

BYTE " \* 2. SPRITE = RS 70 (500 ml) \*",0dh,0ah

BYTE " \* 3. DEW = RS 70 (500 ml) \*",0dh,0ah

BYTE " \* 4. SALAD = RS 60 \*",0dh,0ah

BYTE " \* 5. EXIT \*",0dh,0ah,0

drinksprices DWORD 70,70,70,30

dessert\_menu BYTE " \* 1. CHOCOLATE CAKE = RS 155 \*",0dh,0ah

BYTE " \* 2. GULAAB JAAMUN = RS 145 \*",0dh,0ah

BYTE " \* 3. KHEER = RS 75 \*",0dh,0ah

BYTE " \* 4. ICE-CREAM (ALL FLAVOURS) = RS 100 \*",0dh,0ah

BYTE " \* 5. EXIT \*",0dh,0ah,0

dessertprices DWORD 155,145,75,100

ent\_choice byte "ENTER YOUR CHOICE : ",0

spaces BYTE " ",0

lesspaces BYTE " ",0

error\_prompt BYTE " PLEASE FOLLOW INSTRUCTIONS CORRECTLY ",0dh,0ah,0

error\_title\_prompt BYTE " ERROR",0dh,0ah,0

Quantity BYTE "QUANTITY : ",0

billing BYTE " TOTAL BILL : RS ",0

luckyfactor BYTE "DO YOU WANT TO TRY YOUR LUCK FOR DISCOUNT : ",0dh,0ah,0

luck\_title BYTE "WANNA TRY LUCK?",0dh,0ah,0

discountfactor BYTE 10,20,30

discount DWORD ?

fact DWORD ?

congoprompt BYTE "YOU HAVE WON DISCOUNT OF : ",0

persign BYTE "%",0

ent\_name BYTE " ENTER NAME : ",0

ent\_order BYTE " ENTER ORDER NO : ",0

again\_ent BYTE " ENTER YOUR CHOICE AGAIN : ",0

namepr byte " Name : ",0

ordernum byte " ORDER NO. : ",0

bufSize DWORD ($-cust\_name)

errMsg BYTE "Cannot open file",0dh,0ah,0

billstr byte 20 dup(0)

fmt byte "%d",0

filename byte "ORDER ",20 dup(0)

fileHandle HANDLE ?

filesize dword ?

buff byte BUFFER\_SIZE dup(?)

cust\_name BYTE 50 dup(?)

filetypeee byte ".txt",0

order\_num byte 50 dup(?)

.code

main proc

call crlf

call crlf

mov eax,blue+(white\*16)

call clrscr

call settextcolor

call clrscr

mov edx,OFFSET heading

call writestring

call crlf

call crlf

mov edx,offset ent\_order

call writestring

mov edx,offset order\_num

mov ecx,lengthof order\_num

call readstring

mov edx,offset ent\_name

call writestring

mov edx,offset cust\_name

mov ecx,lengthof cust\_name

call readstring

L1:

call crlf

call crlf

mov edx,OFFSET menu\_list

call writestring

call crlf

call crlf

mov edx,OFFSET spaces

call writestring

mov edx,offset ent\_choice

call writestring

call readdec

mov edx,OFFSET spaces

call writestring

call crlf

call CheckChoice

cmp eax,1

je l2

cmp eax,2

je l3

cmp eax,3

je l4

cmp eax,4

je l5

cmp eax,5

je l6

jmp last

l2:

call breakfast

jmp l7

l3:

call lunch

jmp l7

l4:

call fastfood

jmp l7

l5:

call drinks

l6:

call dessert

l7:

mov edx,OFFSET bill\_confirm

call writestring

mov edx,OFFSET spaces

call writestring

call readdec

call CheckAgainChoice

cmp eax,1

je L1

last:

call crlf

call crlf

mov ebx,bill

cmp ebx,100

jl nodiscount

mov edx,OFFSET luckyfactor

mov ebx,OFFSET luck\_title

call msgboxask

cmp eax,6

jne nodiscount

call randomize

mov eax,3

call randomrange

mov fact,eax

mov esi,OFFSET discountfactor

add esi,fact

mov edx,0

mov dl,BYTE PTR[esi]

push edx

mov eax,bill

mul edx

mov ecx,100

div ecx

mov discount,eax

mov edx,OFFSET spaces

call writestring

mov edx,OFFSET congoprompt

call writestring

mov eax,0

pop edx

mov al,dl

call writedec

mov edx,OFFSET persign

call writestring

call crlf

jmp nodiscount

cmp eax,7

mov discount,0

je nodiscount

nodiscount:

call generateFilename

mov edx,OFFSET ordernum

call writestring

mov edx,OFFSET order\_num

call writestring

call crlf

call crlf

mov edx,OFFSET namepr

call writestring

mov edx,OFFSET cust\_name

call writestring

call crlf

call crlf

mov eax,red+(white\*16)

call settextcolor

mov edx,OFFSET billing

call writestring

mov eax,bill

sub eax,discount

mov bill,eax

call writedec

call crlf

call crlf

call createorderfile

exit

main endp

CheckChoice PROC

l1:

cmp eax,1

jl l2

cmp eax,6

jg l2

jmp last

l2:

mov edx, OFFSET error\_prompt

mov ebx, OFFSET error\_title\_prompt

call msgbox

mov edx, OFFSET again\_ent

call writestring

call readdec

jmp l1

last:

ret

CheckChoice ENDP

breakfast PROC

mov edx,OFFSET breakfast\_menu

call writestring

mov edx,OFFSET spaces

call writestring

call crlf

call crlf

mov edx,OFFSET spaces

call writestring

mov edx,OFFSET ent\_choice

call writestring

call readdec

call crlf

mov edx,OFFSET spaces

call writestring

call CheckChoiceBreakFast

cmp eax,1

je choice1

cmp eax,2

je choice2

cmp eax,3

je choice3

cmp eax,4

je choice4

cmp eax,5

jmp last

choice1:

mov edx,OFFSET Quantity

call writestring

call readdec

mov ecx,eax

mov ebx,[breakfastprices]

billadd1:

add bill,ebx

loop billadd1

jmp last

choice2:

mov edx,OFFSET Quantity

call writestring

call readdec

mov ecx,eax

mov ebx,[breakfastprices + 4]

billadd2:

add bill,ebx

loop billadd2

jmp last

choice3:

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

mov ebx,[breakfastprices + 8]

billadd3:

add bill,ebx

loop billadd3

jmp last

choice4:

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

mov ebx,[breakfastprices + 12]

addbill4:

add bill,ebx

loop addbill4

last:

ret

breakfast ENDP

CheckChoiceBreakFast PROC

l1:

cmp eax,1

jl l2

cmp eax,5

jg l2

jmp last

l2:

mov edx, OFFSET error\_prompt

mov ebx, OFFSET error\_title\_prompt

call msgbox

mov edx, OFFSET again\_ent

call writestring

call readdec

jmp L1

last:

ret

CheckChoiceBreakFast ENDP

CheckAgainChoice PROC

l1:

cmp eax,1

jl l2

cmp eax,2

jg l2

jmp last

l2:

mov edx, OFFSET error\_prompt

mov ebx, OFFSET error\_title\_prompt

call msgbox

mov edx, OFFSET again\_ent

call writestring

call readdec

jmp l1

last:

ret

CheckAgainChoice ENDP

lunch PROC

mov edx,OFFSET lunch\_menu

call writestring

mov edx,OFFSET spaces

call writestring

call crlf

call crlf

mov edx,OFFSET spaces

call writestring

mov edx,offset ent\_choice

call writestring

call readdec

call crlf

mov edx,OFFSET spaces

call writestring

call CheckChoiceLunch

cmp eax,1

je choice1

cmp eax,2

je choice2

cmp eax,3

je choice3

cmp eax,4

je choice4

cmp eax,5

jmp last

choice1:

mov edx,OFFSET Quantity ;HEREEEEEEEEEEEEEEEEEEEEEEE

call writestring

call readdec

mov ecx,eax

mov ebx,[lunchprices]

addbill1:

add bill,ebx

loop addbill1

jmp last

choice2:

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

mov ebx,[lunchprices + 4]

addbill2:

add bill,ebx

loop addbill2

call bread

jmp last

choice3:

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

mov ebx,[lunchprices + 8]

addbill3:

add bill,ebx

loop addbill3

call bread

jmp last

choice4:

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

mov ebx,[lunchprices + 12]

addbill4:

add bill,ebx

loop addbill4

last:

ret

lunch ENDP

CheckChoiceLunch PROC

l1:

cmp eax,1

jl l2

cmp eax,5

jg l2

jmp last

l2:

mov edx, OFFSET error\_prompt

mov ebx, OFFSET error\_title\_prompt

call msgbox

mov edx, OFFSET again\_ent

call writestring

call readdec

jmp L1

last:

ret

CheckChoiceLunch ENDP

bread PROC

mov edx,OFFSET bread\_menu

call writestring

call crlf

call crlf

mov edx,OFFSET spaces

call writestring

mov edx,OFFSET ent\_choice

call writestring

call readdec

mov edx,OFFSET spaces

call writestring

call CheckChoiceBread

call crlf

mov edx,OFFSET spaces

call writestring

cmp eax,1

je choice1

cmp eax,2

je choice2

jmp last

choice1:

mov ebx,[breadprices]

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

addbill1:

add bill,ebx

loop addbill1

jmp last

choice2:

mov ebx,[breadprices + 4]

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

addbill2:

add bill,ebx

loop addbill2

last:

ret

bread ENDP

CheckChoiceBread PROC

l1:

cmp eax,1

jl l2

cmp eax,3

jg l2

jmp last

l2:

mov edx, OFFSET error\_prompt

mov ebx, OFFSET error\_title\_prompt

call msgbox

mov edx, OFFSET again\_ent

call writestring

call readdec

jmp L1

last:

ret

CheckChoiceBread ENDP

fastfood PROC

mov edx,OFFSET fastfood\_menu

call writestring

call crlf

mov edx,OFFSET spaces

call writestring

mov edx,OFFSET ent\_choice

call writestring

call readdec

call crlf

mov edx,OFFSET spaces

call writestring

call CheckChoiceFastfood

cmp eax,1

je choice1

cmp eax,2

je choice2

cmp eax,3

je choice3

cmp eax,4

je choice4

cmp eax,5

jmp last

choice1:

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

mov ebx,[fastfoodprices]

addbill1:

add bill,ebx

loop addbill1

jmp last

choice2:

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

mov ebx,[fastfoodprices + 4]

addbill2:

add bill,ebx

loop addbill2

jmp last

choice3:

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

mov ebx,[fastfoodprices + 8]

addbill3:

add bill,ebx

loop addbill3

jmp last

choice4:

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

mov ebx,[fastfoodprices + 12]

L44:

add bill,ebx

loop L44

last:

ret

fastfood ENDP

CheckChoiceFastfood PROC

l1:

cmp eax,1

jl l2

cmp eax,5

jg l2

jmp last

l2:

mov edx, OFFSET error\_prompt

mov ebx, OFFSET error\_title\_prompt

call msgbox

mov edx, OFFSET again\_ent

call writestring

call readdec

jmp L1

last:

ret

CheckChoiceFastfood ENDP

drinks PROC

mov edx,OFFSET drinks\_menu

call writestring

call crlf

mov edx,OFFSET spaces

call writestring

mov edx,OFFSET ent\_choice

call writestring

call readdec

call crlf

mov edx,OFFSET spaces

call writestring

call CheckChoiceDrink

cmp eax,1

je choice1

cmp eax,2

je choice2

cmp eax,3

je choice3

cmp eax,4

je choice4

cmp eax,5

jmp last

choice1:

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

mov ebx,[drinksprices]

addbill1:

add bill,ebx

loop addbill1

jmp last

choice2:

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

mov ebx,[drinksprices + 4]

addbill2:

add bill,ebx

loop addbill2

jmp last

choice3:

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

mov ebx,[drinksprices + 8]

addbill3:

add bill,ebx

loop addbill3

jmp last

choice4:

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

mov ebx,[drinksprices + 12]

addbill4:

add bill,ebx

loop addbill4

last:

ret

drinks ENDP

CheckChoiceDrink PROC

l1:

cmp eax,1

jl l2

cmp eax,5

jg l2

jmp last

l2:

mov edx, OFFSET error\_prompt

mov ebx, OFFSET error\_title\_prompt

call msgbox

mov edx, OFFSET again\_ent

call writestring

call readdec

jmp L1

last:

ret

CheckChoiceDrink ENDP

Dessert PROC

mov edx,OFFSET dessert\_menu

call writestring

call crlf

mov edx,OFFSET spaces

call writestring

mov edx,OFFSET ent\_choice

call writestring

call readdec

call crlf

mov edx,OFFSET spaces

call writestring

call CheckChoiceDessert

cmp eax,1

je choice1

cmp eax,2

je choice2

cmp eax,3

je choice3

cmp eax,4

je choice4

cmp eax,5

jmp last

choice1:

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

mov ebx,[dessertprices]

addbill1:

add bill,ebx

loop addbill1

jmp last

choice2:

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

mov ebx,[dessertprices + 4]

addbill2:

add bill,ebx

loop addbill2

jmp last

choice3:

mov edx,OFFSET Quantity

call writestring

call readdec

mov ecx,eax

call crlf

mov ebx,[dessertprices + 8]

addbill3:

add bill,ebx

loop addbill3

jmp last

choice4:

mov edx,OFFSET Quantity

call writestring

call readdec

call crlf

mov ecx,eax

mov ebx,[dessertprices + 12]

addbill4:

add bill,ebx

loop addbill4

last:

ret

Dessert ENDP

CheckChoiceDessert PROC

l1:

cmp eax,1

jl l2

cmp eax,5

jg l2

jmp last

l2:

mov edx, OFFSET error\_prompt

mov ebx, OFFSET error\_title\_prompt

call msgbox

mov edx, OFFSET again\_ent

call writestring

call readdec

jmp L1

last:

ret

CheckChoiceDessert ENDP

generatefilename proc

mov edi, offset filename ; target string

mov esi, offset order\_num ; source string

call str\_concat

ret

generatefilename endp

Str\_concat PROC

xor eax, eax

LScanString:

scasb

jne LScanString

dec edi

mov ecx, 1

cld

LConcatStrings:

inc ecx

rep movsb

cmp [esi], eax

jne LConcatStrings

xor eax, eax

mov edi,offset filename

mov esi,offset filetypeee

LScanString2:

scasb

jne LScanString2

dec edi

mov ecx, 1

cld

LConcatStrings2:

inc ecx

rep movsb

cmp [esi], eax

jne LConcatStrings2

LExitStr\_concat:

ret

Str\_concat ENDP

CreateOrderFile PROC

mov edx,offset filename

call openinputfile

mov fileHandle,eax

mov edx,offset buff

mov ecx,sizeof buff

call readfromfile

mov filesize,eax

mov buff[eax],0DH

inc eax

inc filesize

mov eax,filehandle

call closefile

mov eax,filesize

mov esi,offset buff

add esi,eax

mov ebx,0

mov ecx,lengthof billheader

mov edi,offset billheader

Lheader:

mov bl,[edi]

mov [esi],bl

inc esi

inc edi

inc filesize

loop Lheader

mov edx,0DH

mov [esi],edx

inc esi

inc filesize

mov ecx,lengthof orderNum

mov edi,offset orderNum

LOrderNum:

mov bl,[edi]

mov [esi],bl

inc esi

inc edi

inc filesize

loop LOrderNum

mov ecx,lengthof lesspaces

mov edi,offset lesspaces

Lspace:

mov bl,[edi]

mov [esi],bl

inc esi

inc edi

inc filesize

loop Lspace

mov ecx,lengthof order\_num

mov edi,offset order\_num

L1:

mov bl,[edi]

mov [esi],bl

inc esi

inc edi

inc filesize

loop L1

mov edx,0DH

mov [esi],edx

inc esi

inc filesize

mov ecx,lengthof namepr

mov edi,offset namepr

Lnamepr:

mov bl,[edi]

mov [esi],bl

inc esi

inc edi

inc filesize

loop Lnamepr

mov ecx,lengthof lesspaces

mov edi,offset lesspaces

Lspace2:

mov bl,[edi]

mov [esi],bl

inc esi

inc edi

inc filesize

loop Lspace2

mov ecx,lengthof cust\_name

mov edi,offset cust\_name

L2:

mov bl,[edi]

mov [esi],bl

inc esi

inc edi

inc filesize

loop L2

mov edx,0DH

mov [esi],edx

inc esi

inc filesize

push bill

push offset fmt

push offset billstr

call wsprintf

mov ecx,lengthof billing

mov edi,offset billing

Lbilling:

mov bl,[edi]

mov [esi],bl

inc esi

inc edi

inc filesize

loop Lbilling

mov ecx,lengthof lesspaces

mov edi,offset lesspaces

Lspace3:

mov bl,[edi]

mov [esi],bl

inc esi

inc edi

inc filesize

loop Lspace3

mov ecx,lengthof billstr

mov edi,offset billstr

L3:

mov bl,[edi]

mov [esi],bl

inc esi

inc edi

inc filesize

loop L3

mov edx,offset filename

call createoutputfile

mov filehandle,eax

mov edx,offset buff

mov ecx,filesize

call writetofile

mov eax,filehandle

call closefile

add esp,12

ret

CreateOrderFile endp

end main

CONCLUSIONS:

The project concludes with solving the issue of the restaurant with prior manual system and provide a proper new automated system with better handling of the orders.

REFRENCES:

The project has been built without any reference. However little help was taking for filing, from the book and guidance in implementation of some Irvine functions was taken from:

* [Irvine library manual](https://csc.csudh.edu/mmccullough/asm/help/index.html?page=source%2Fwin32lib%2Fsetfilepointer.htm)

And some errors which occurred while coding were solved by taking and idea from:

* Online forums

14