PURCHASE ORDER GENERATION USING ASSEMBLY LANGUAGE

Report submitted for Review-3 of

Microprocessor and Interfacing By:

Sr. No	Name	Registration Number
1.	Archit Thakur	18BCE0480
2.	Neel Choksi	19BCE0990
3.	Adabala Shanmukha Charan	19BCE2059
4.	Mangamudi Yaswanth	19BCE0030

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Code: CSE2006

Course Name: Microprocessor and Interfacing

Professor: ANTONY XAVIER GLITTAS X

Slot: L13+L14

DECLARATION

We hereby declare that the thesis entitled, “Bill Order Generation System" submitted by us, for the award of the degree of Bachelor of Technology in Computer Science to VIT is a record of bonafide work carried out by us under the supervision of Prof. ANTONY XAVIER GLITTAS X

We further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Vellore
Date: 10 December 2021

TABLE OF CONTENT

1. INTRODUCTION

Purchase order is a document generated by the purchasing department for buying goods from a seller or a supplier.This document includes the types of goods, proposed price, quantity and expected date of delivery.[13]

Purchase order is important to keep track of the spending of the business on the material or goods purchased.Before generation of the purchase order , a tender is generated by the buyer to get the proposed unit prices of the required products from different suppliers. Binding Contract is a document created after the seller accepts the purchase order to resolve any disputes between the buyer and the seller[2].In this project a Tender(Request for quotation), Purchase Order Generation along with Binding Contract Generation software will be written for a 8086 microprocessor .

2. OBJECTIVE

The objective of this project is to incorporate all the concepts used and learnt under the course “Microprocessor and Interfacing” and create a real life problem solving application using 8086 Programming. We have made use of our programming skills and knowledge to build a Purchase Order generation system that generates purchase orders, Binding contracts of various formats and Tenders for the user’s input.

3.MOTIVATION

The order generation process at a distributor level starts with the purchase order from the buyer. The suppliers need to have a purchase order to create a sales order and invoice. The tender is an important document for the buyer since it enables the buyer to get different quotations for the required products from various suppliers. After receiving the unit prices from the suppliers, the buyer generates a purchase order which helps keep record of the spendings of the company. The buyer could have different requirements related to delivery of goods , payments, delivery address and billing addresses which are mentioned in the binding contract and sent to the supplier along with the binding contract.The motivation behind building this project was to build a process for the buyer to generate a purchase order or purchase order with a tender , binding contract in different situations so that the buyer can get on time delivery of the required goods at the best price. A proper flow for tender , purchase order, binding contract even helps the supplier to provide goods, receive payments and make delivery according to the requirements of the buyer.

4. PROPOSED WORK AND IMPLEMENTATION

4.1 METHODOLOGY

The project is a menu based program, which takes input from the user and user can select between Generating a purchase order, or a purchase order with binding contract and tender.The user presses 1 for Purchase order generation and 2 for purchase order with binding contract and tender.When user presses 1, the user is asked for following inputs:

1) Number of items
2) Quantity of item
3) Price of item

We make use of interrupts to take input from the user and then store it in 2 Arrays, For quantity and price of each item. We make one additional array which multiplies both Quantity and price and then it is stored in the 3rd (additional array).Now we format the data and display it in a neat manner, showing the corresponding total For the items and the total cost which has been calculated.We have made use of SHR and AND operations to display our total price and all the prices Which are in 2 digits (since 8086 allows only 1 character to be displayed from an interrupt).

When the user chooses option 2 , the code is directed for generating a tender, purchase order with binding contract using switch case logic. First the user is prompted to input details for request for quotation.To display the statement to take the input, print macro is developed to print the statement entered into the macro.Input is taken by developing an input macro using the configuration to take an input using int 21h interrupt.

Request for quotation involves the following inputs:

1. quotation number
2. vendor name
3. vendor reference id
4. buyer company name
5. date of tender
6. number of products
7. name of each product
8. product description for each product
9. quantity for each product.

The quotation number, vendor name, vendor reference id, buyer company name, date of tender, number of products are stored in their respective variables whereas the product description , product name, quantity are stored in three different arrays. The arrays are accessed using the offset address from the array variables and each item is printed using the print macro taking in the offset address of the variable. The variables are also printed using the print macro developed which takes in the offset address of the variable to print to the screen . The newline and space macros are generated to add newlines and spaces wherever required to format the document. Tender is displayed on the screen using the inputs for request for quotation.

Inputs for purchase order are taken involving the unit prices for the products given by the supplier. The inputs are stored in variables and unit prices are stored in the unit price array. The inputs for purchase involve:

1. unit price for each product
2. PO(Purchase Order) number
3. expected delivery date
4. buyer phone number
5. buyer email address
6. buyer company address
7. vendor name
8. vendor address
9. buyer ship to address
10. buyer requisitioner name
11. ship to buyer via
12. shipping terms

The product amount is calculated by multiplying the quantity of each product by the unit price and gets stored in an amount array.The total amount is calculated by adding the elements of the amount array.Using the total amount, input variables for purchase order and product description array , product name array , quantity array, unit price array , the purchase order document is displayed on the screen.

The user is further displayed the 8 types of binding contract available to generate. Based on the choice of the user , the user is asked for the specific inputs to generate a binding contract. This is done using switch case logic.The 8 types of binding contract involve the following inputs and display the binding contract accordingly.

Type 1: Send PO to the supplier under normal circumstances.
Type 2: Send PO to supplier , also informed supplier over the phone.
Type 3: Send PO to supplier, also requesting supplier to make early delivery.
Type 4: Send PO to supplier , also requesting supplier to do partial delivery.
Inputs : Item numbers as comma separated values that are included in the partial delivery.

Type 5: Send PO to supplier , also requesting supplier to send proforma invoice to submit advance payment.
Inputs: Advance payment percentages including percent of advance payment , percent of payment after shipment and percent of payment after delivery.

Type 6: Send PO to supplier, also informing supplier that the delivery address and billing address is different.
Inputs: States names for delivery and invoicing.

Type 7: Send PO to supplier, informing if the item's delivery to be done every month.
Inputs: number of pieces to be sent annually and number of pieces to be sent monthly.

Type 8: Send PO to supplier, informing if the item's delivery to be done every month.
Inputs: number of pieces to be sent annually and number of pieces to be sent quarterly.

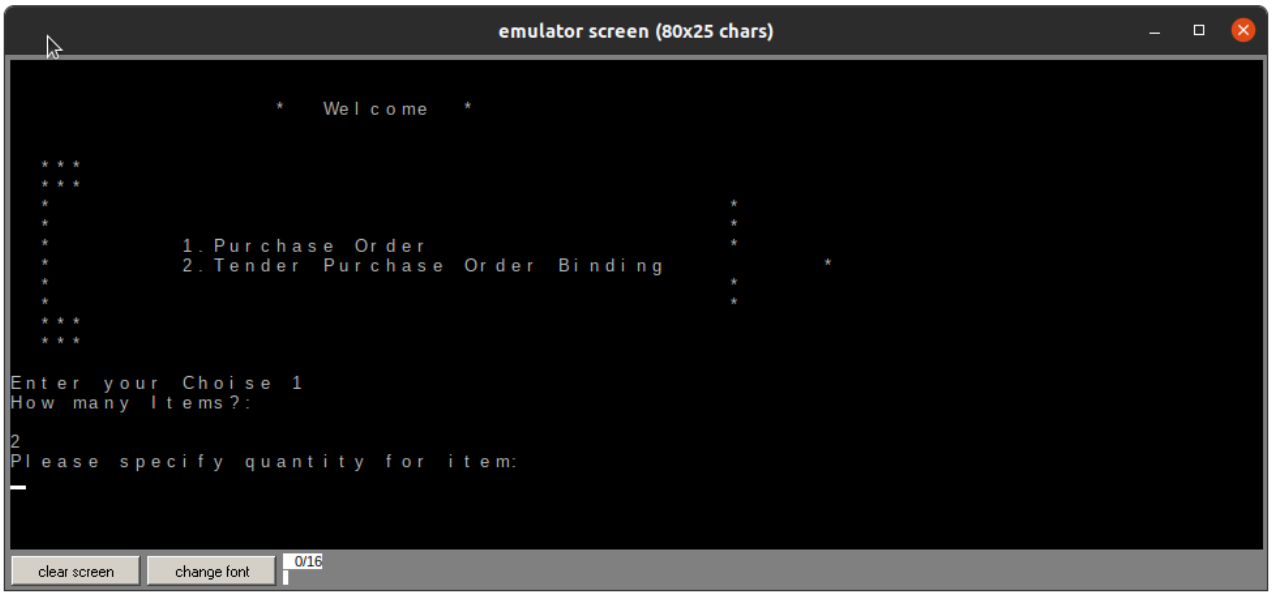
After the display of binding contract according to user choice, the user is shown the main menu again to generate purchase orders or purchase orders with tender and binding contract for next orders.

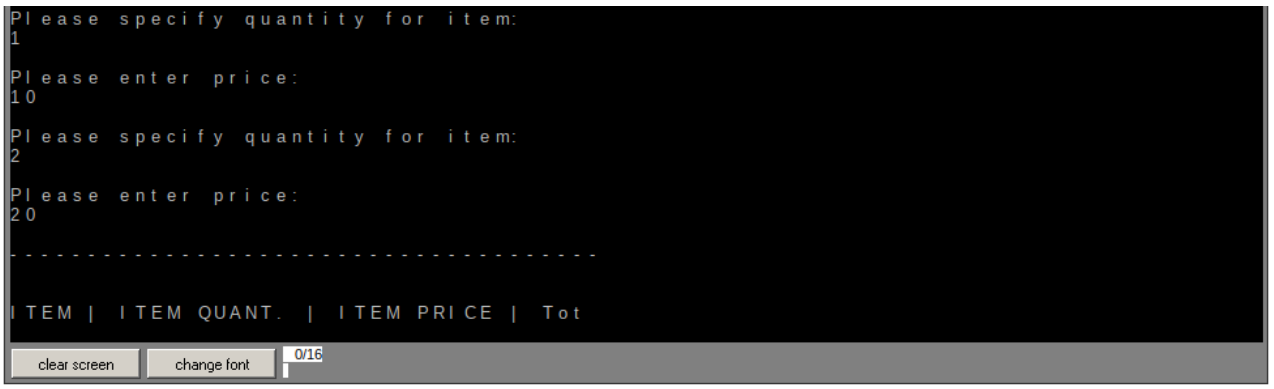
4.2 TOOLS USED

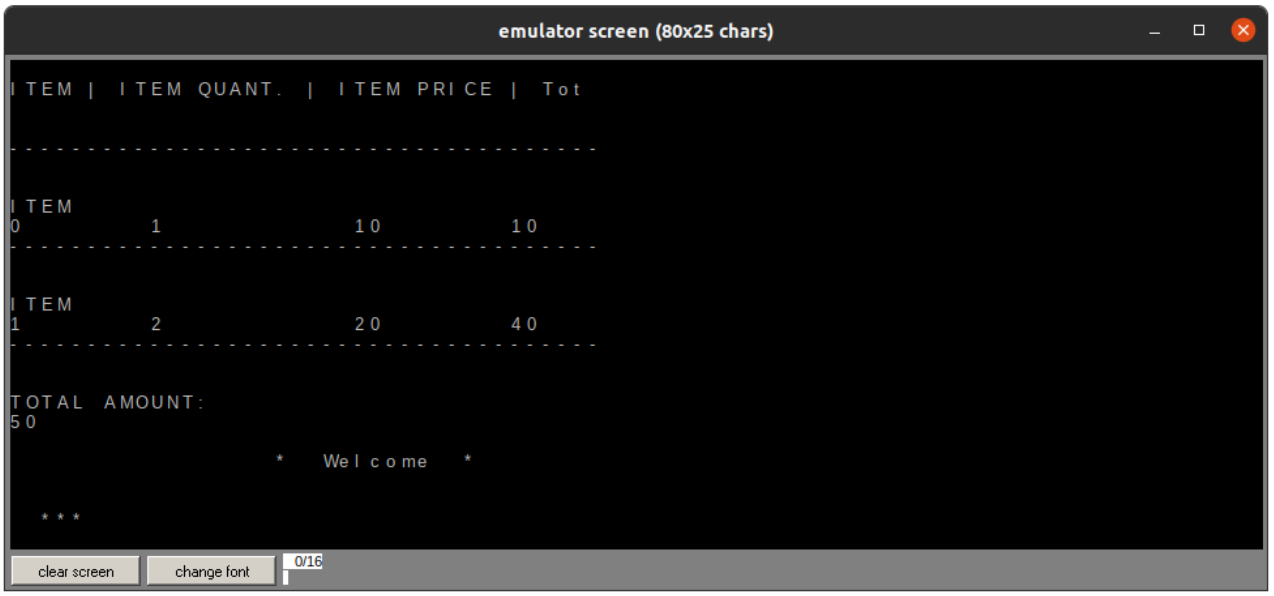
The software used for this project involves an emu8086 emulator on a laptop with minimum of 4GB RAM and 256GB of storage.

4.3 SCREENSHOT AND DEMO ALONG WITH VISUALIZATION

Purchase Order generation for 2 items with quantity 1 and 2 respectively and price 10 and 20 respectively.

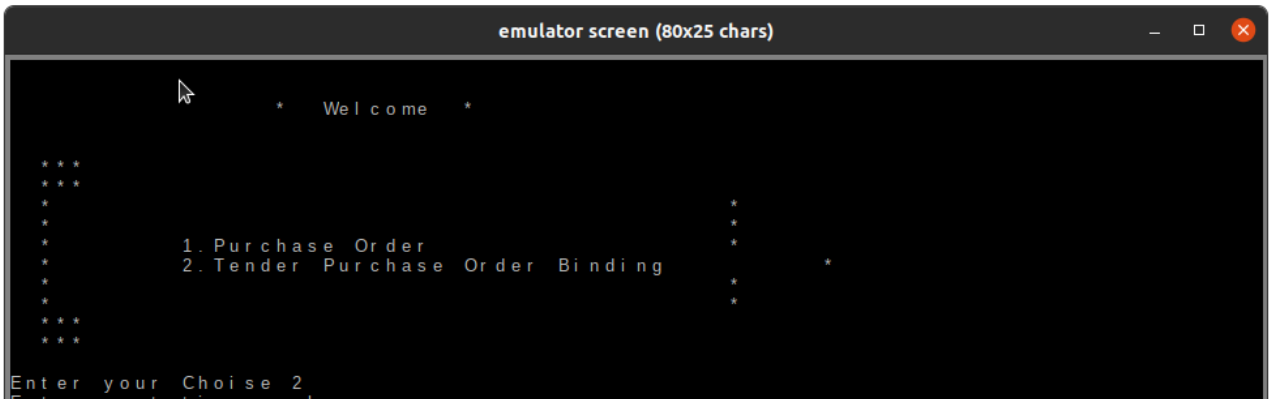






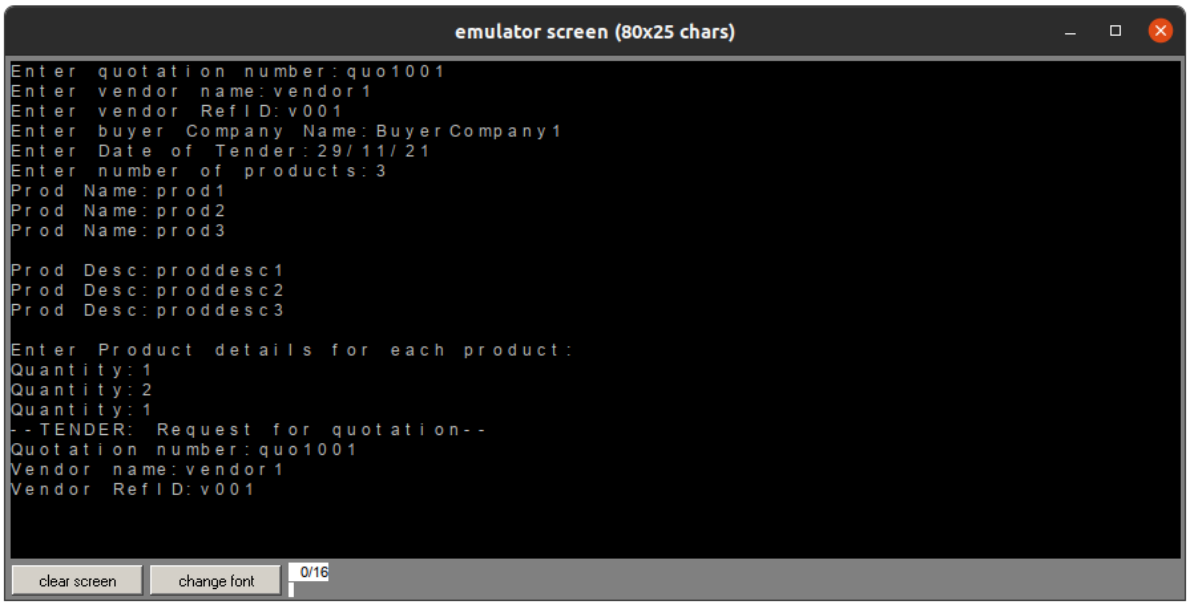
The menu screen is displayed again after completion of either of the two processes.

Choice 2 is selected.

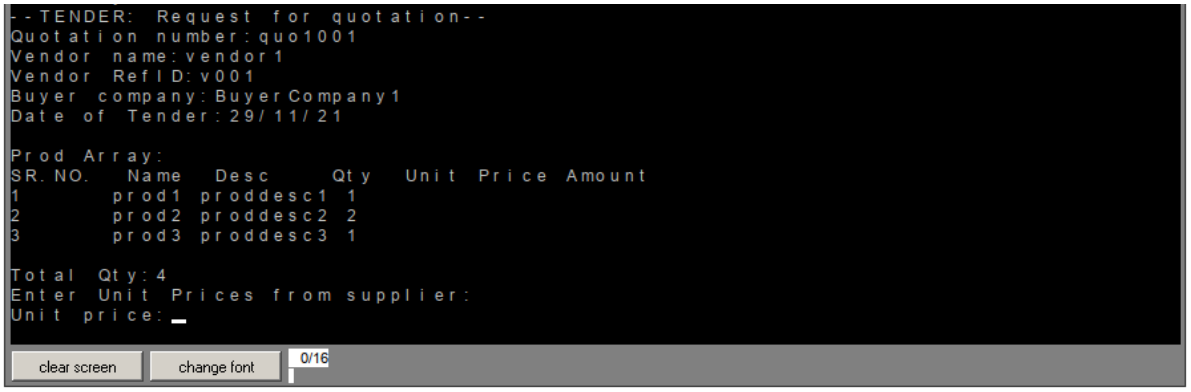


Tender Request for Quotation , Purchase Order, Binding Contract Generation

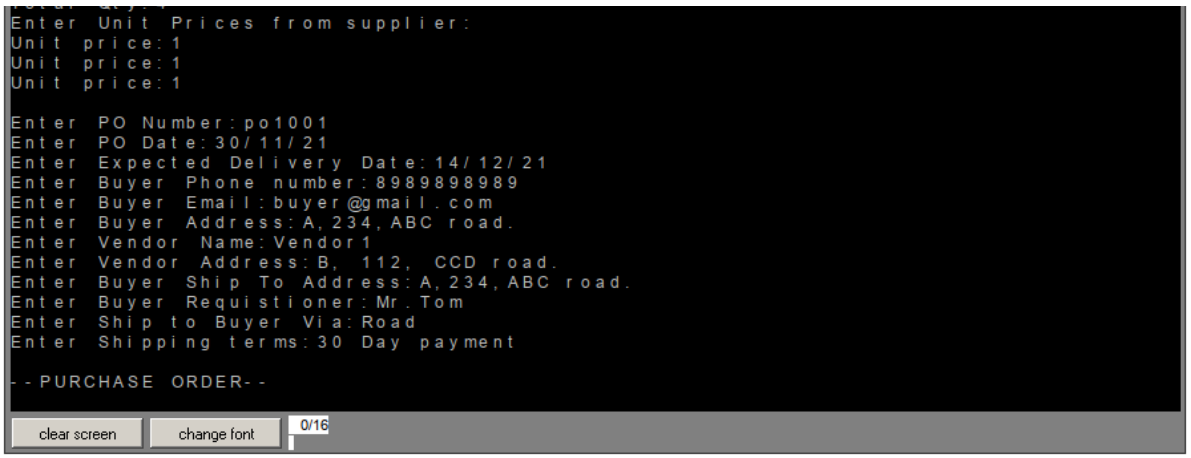
Request for quotation inputs



Tender displayed to the screen



Inputs for Purchase Order generation

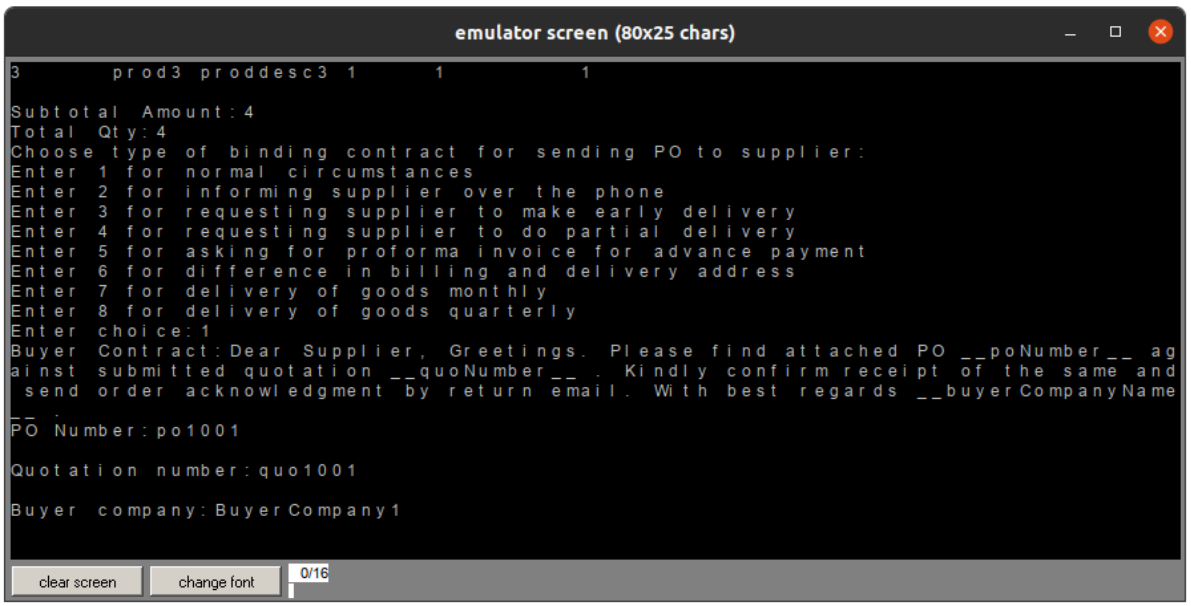


Purchase order displayed to the screen

```
--PURCHASE ORDER--
Quotation number:quo1001
PO Number:po1001
PO Date:30/11/21
Expected Delivery Date:14/12/21
Buyer company:BuyerCompany1
Buyer Phone Number:8989898989
Buyer Email:buyer@gmail.com
Buyer Address:A,234,ABC road.
Vendor Name:Vendor1
Vendor Address:B, 112, CCD road.
Buyer Ship To Address:A,234,ABC road.
Buyer Requistioner:Mr.Tom
Ship to buyer via:Road
Shipping Terms:30 Day payment
```

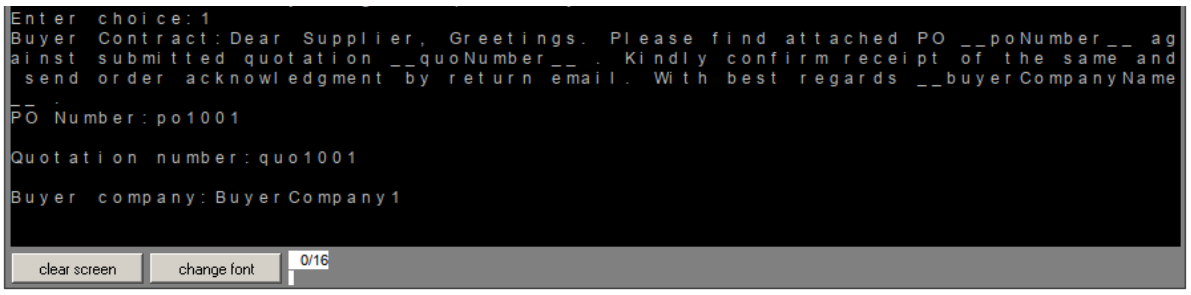
```
Prod Array:
SR.NO.   Name   Desc     Qty   Unit Price Amount
1        prod1 proddesc1 1      1          1
2        prod2 proddesc2 2      1          2
3        prod3 proddesc3 1      1          1

Subtotal Amount:4
Total Qty:4
```

Input for type of binding contract to be generated and generation of binding contract accordingly

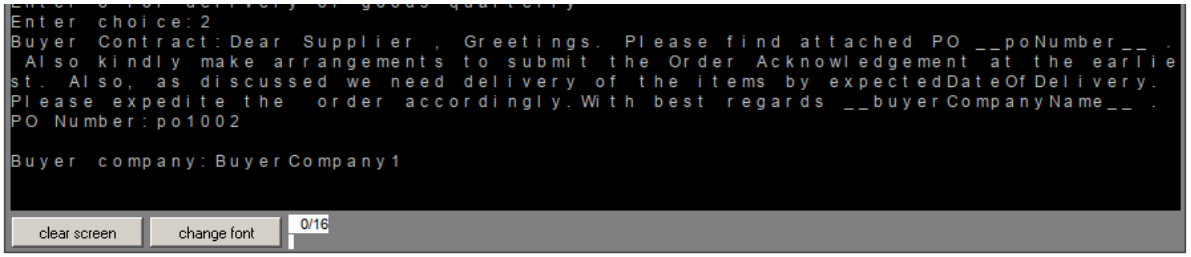


There is provision made to generate 8 different types of binding contact:

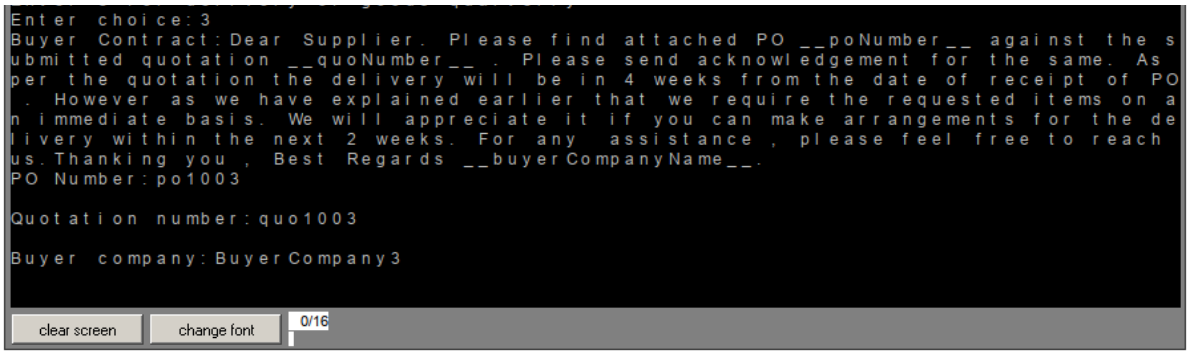
Type 1: Send PO to the supplier under normal circumstances.



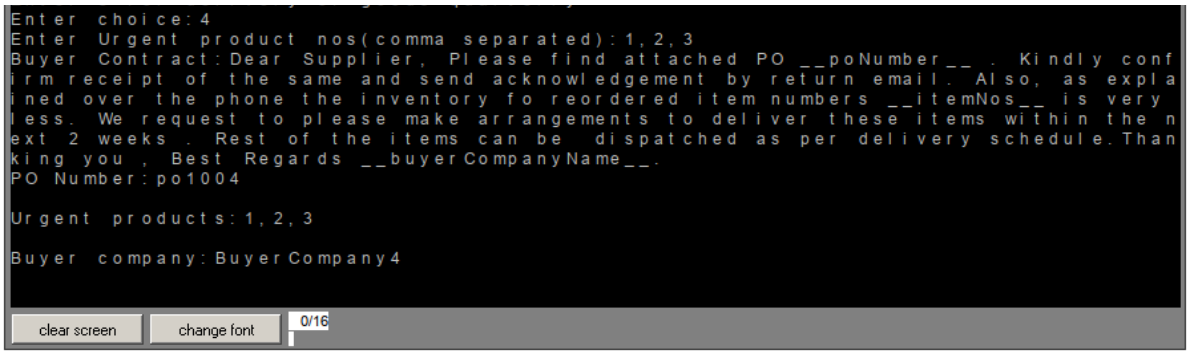
Type 2: Send PO to supplier , also informed supplier over the phone.



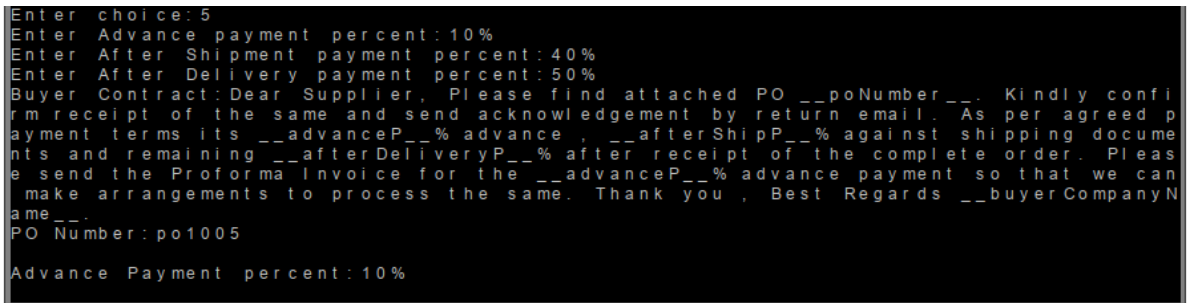
Type 3: Send PO to supplier, also requesting supplier to make early delivery.

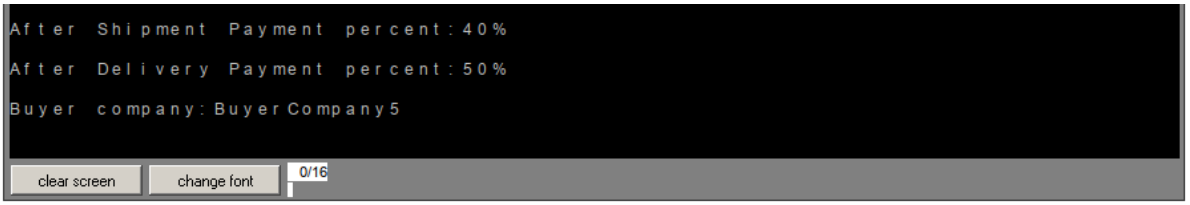


Type 4: Send PO to supplier , also requesting supplier to do partial delivery.
Number of items to do partial delivery of are taken input as comma separated string:
__itemNos__

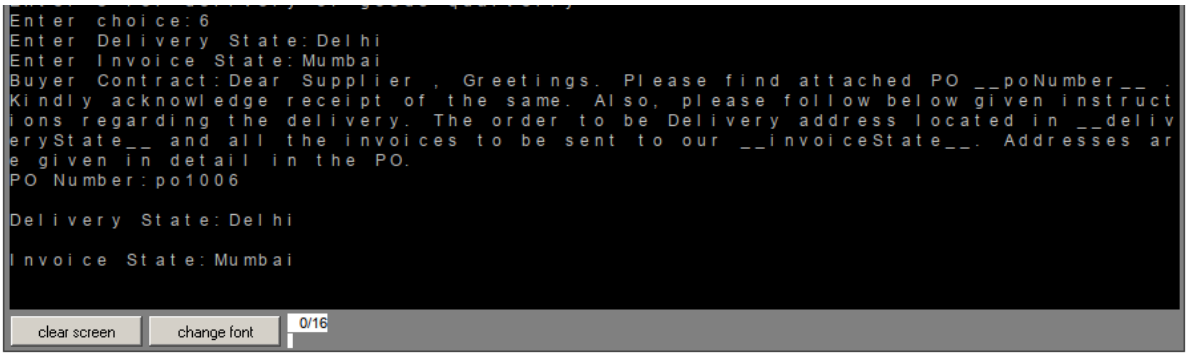


Type 5: Send PO to supplier , also requesting supplier to send proforma invoice to submit advance payment.
Advance payment percent inputs taken:
Advance percent : __advanceP__
After shipping: __afterShipP__
After delivery: __afterDeliveryP__

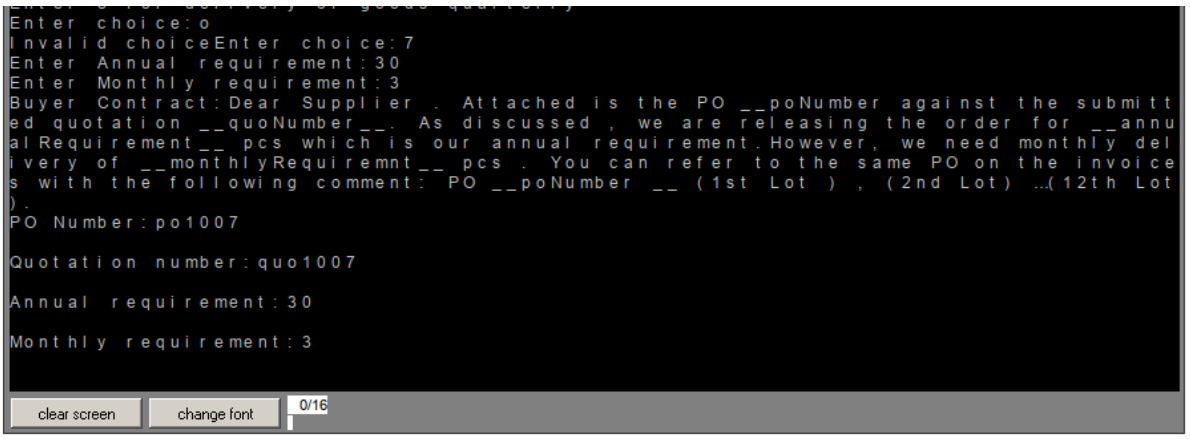




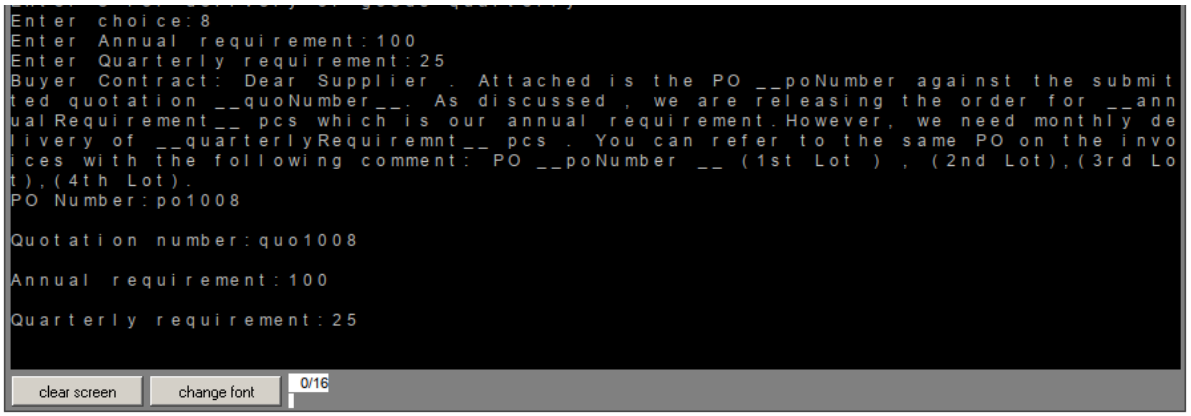
Type 6: Send PO to supplier, also informing supplier that the delivery address and billing address is different.
Inputs taken to display different invoice and delivery state:
Delivery state : __deliveryState__
Invoice state: __invoiceState__



Type 7: Send PO to supplier, informing if the item's delivery to be done every month.
Number of pieces inputs taken for annual and monthly requirement:
Annual requirement : __annualRequirement__
Monthly requirement : __monthlyRequiremnt__



Type 8: Send PO to supplier, informing if the item's delivery to be done every month.
Number of pieces inputs taken for annual and quarterly requirement:
Annual requirement : __annualRequirement__
Quarterly requirement : __quarterlyRequiremnt__


Enter choice:8
Enter Annual requirement:100
Enter Quarterly requirement:25
Buyer Contract: Dear Supplier . Attached is the PO __poNumber against the submitted quotation __quoNumber__. As discussed , we are releasing the order for __annualRequirement__ pcs which is our annual requirement.However, we need monthly delivery of __quarterlyRequiremnt__ pcs . You can refer to the same PO on the invoices with the following comment: PO __poNumber __ (1st Lot) , (2nd Lot),(3rd Lot),(4th Lot).
PO Number:po1008
Quotation number:quo1008
Annual requirement:100
Quarterly requirement:25
clear screen
change font
0/16

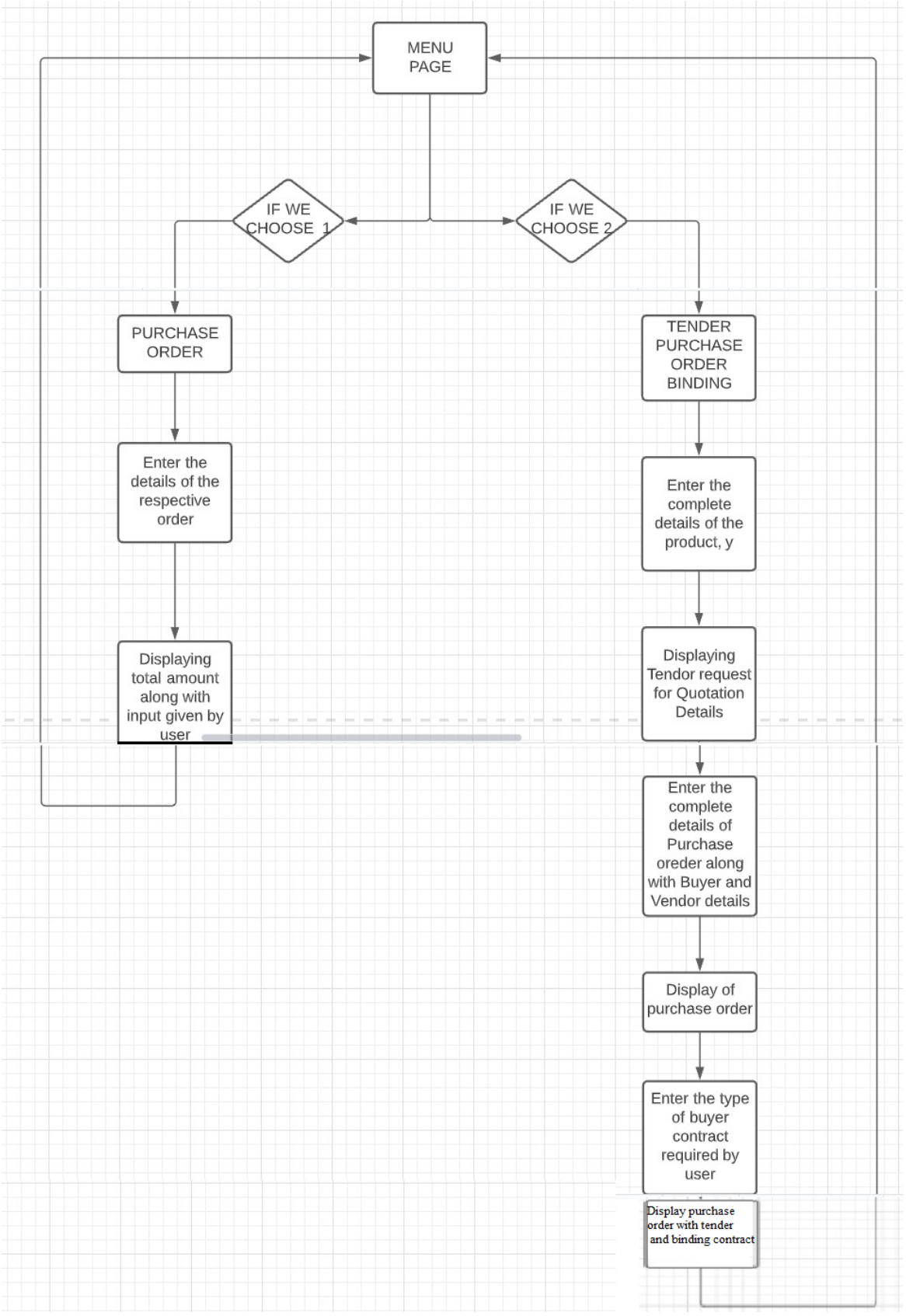
5. RESULTS AND DISCUSSIONS

The user is given the choice to create only a purchase order or to create a tender , purchase order with a binding contract. The user is directed to the part of the code based on the choice selected. A menu is displayed to portray the two different choices and asks the user to choose one option. If the user selects the process of creating the purchase order , he/she is asked for the total number of products, then for each product price and quantity inputs are taken ,finally the user is displayed the purchase order displaying the item number , quantity, unit price amount and at the end a total amount is displayed.

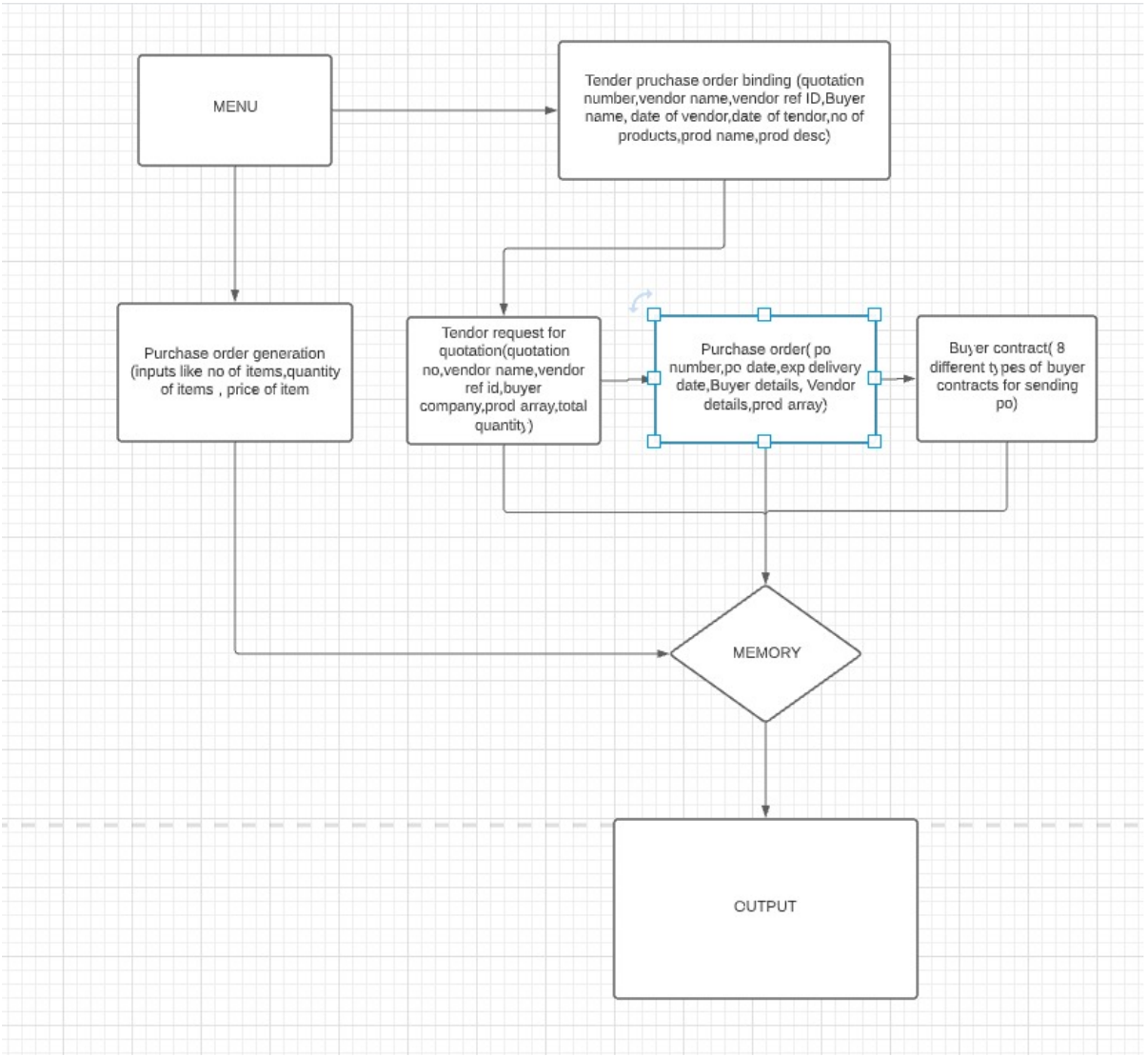
When the user chooses to create a tender with purchase order and binding contract. The user is asked to input the general details to create a request of quotation, product description with the product name and quantity. Based on these details a tender is generated which can be used to display the requirements of the user(buyer) to the vendors. After getting the proposed unit prices from different vendors, the user(buyer) can input unit prices with the general details required for the purchase order which will lead to the generation of a purchase order. After that the user will be asked to select a type of binding contract required . The scenarios for binding contract covered in this project involve a normal binding contract along with sending the purchase order to the vendor , informing the supplier over the phone , requesting the supplier to make early delivery , requesting supplier to do partial delivery, asking the proforma invoice from the supplier for making an advance payment, informing if there is a difference between billing address and delivery address, informing the details to make the delivery of the goods monthly , informing for delivery of the goods quarterly. In each binding contract , specific inputs are taken to enhance the binding contract statement . The specific inputs for binding contract, tender(request for quotation) and purchase order are shown in the section 4.3 Screenshot And Demo Along With Visualization.

6. FLOWCHART ARCHITECTURE DIAGRAM

Flowchart



Architectural Diagram



7. CONCLUSION AND FUTURE WORK

7.1 CONCLUSION

In this project we successfully built a Purchase order generation system that generates Purchase Order , Binding Order of various formats and Tender for the user input.

7.2 LIMITATIONS

The limitation of this project is that the document printed on the terminal window is of the size of the terminal window that is 80 x 20 characters. The terminal can only show a part of the document at a particular time.Also the total number of products is limited to 9.

7.3 SCOPE FOR FUTURE WORK

This code could be deployed on hardware with a display , supporting 8086 assembly instructions. The hardware could be connected to a printer which would print the output on the terminal .The hardware could be embedded with a WiFi module which would help transfer the output data to a database , resulting in a complete Purchase Order Management System for the buyer.The code could be modified to append the purchase order and binding contract to a document using file write commands.

8. REFERENCES

[1]Assembly Code - an overview | ScienceDirect Topics.https://www.sciencedirect.com/topics/computer-science/assembly-code

[2]Purchase Order Process | The End-to-End PO Process in a Nutshell.https://kissflow.com/procurement/purchase-order/purchase-order-process/

[3]Microprocessor - 8086 Instruction Sets.https://www.tutorialspoint.com/microprocessor/microprocessor_8086_instruction_sets.htm

[4]Steps to create a purchase order in SAP r/3 | SAP Community.https://answers.sap.com/questions/1758795/steps-to-create-a-purchase-order-in-sap-r3.html

[5]Assembly programming and 8086 microprocessor by D.S.Jones Oxford university Press inc , united states .

[6]8086 Data Transfer Instructions - Assembly Language Programming. https://microcontrollerslab.com/8086-data-transfer-instructions-assembly-language-programming/

[7]Instruction Set of 8086 Dr.C.Bennila Thangammal, Associate Professor / ECE, RMDEC

[8]PC Assembly Language Paul A. Carter November 11, 2003

[9]John Uffenbeck, “The 8086/88 Family: Design, Programming & Interfacing”, PHI,

[10]Liu, Gibson, “Microcomputer Systems: The 8086/88 Family”, 2nd Edition, PHI,2005.

[11]Know Assembly Language Programming of 8086.https://www.elprocus.com/8086-assembly-language-programs-explanation/

[12]8086 Microprocessor Assembly language programming - ppt download.https://slideplayer.com/slide/9760280/

[13] Purchase Order - Definition, Benefits, How Suppliers Use POs .https://corporatefinanceinstitute.com/resources/knowledge/other/purchase-order/

[14] Purchase Order Template .https://www.vertex42.com/ExcelTemplates/excel-purchase-order.html

[15] Freight On Board - Understanding How FOB Works in Shipping . https://corporatefinanceinstitute.com/resources/knowledge/other/freight-on-board-fob/

9. CODE

```
print macro p1
mov dx,offset p1
mov ah,09h
int 21h
endm

newline macro
mov dl,10
mov ah,02h
int 21h

mov dl,13
mov ah,02h
int 21h

endm

space macro
mov dl,32
mov ah,02h
int 21h
endm

inputProdName macro
 newline
 mov cl,numberOfProds
 loopProd:
   lea dx,prodNameInputDisplay
   mov ah,9
   int 21h

   mov ah,0Ah
   mov dx,offset prodNameBuffer
```

```
  int 21h

  mov si,offset prodNameBuffer + 1
  mov bl,[si]
  mov bh,0
  inc bx
  add si,bx
  mov al,'$'
  mov [si],al

  mov di, offset prodName
  add di, inputIndexProdName
  add inputIndexProdName,10

  mov si, offset prodNameBuffer

  more2:
   mov al,[si]
   mov [di],al
   inc si
   inc di
   cmp al,"$"
   jne more2

  newline
  loop loopProd


endm



inputProdDesc macro
 newline
```

```
mov cl,numberOfProds
loopProd1:
 lea dx,prodDescInputDisplay
 mov ah,9
 int 21h

 mov ah,0Ah
 mov dx,offset prodDescBuffer
 int 21h

 mov si,offset prodDescBuffer + 1
 mov bl,[si]
 mov bh,0
 inc bx
 add si,bx
 mov al,'$'
 mov [si],al

 mov di, offset prodDesc
 add di, inputIndexProdDesc
 add inputIndexProdDesc,25

 mov si, offset prodDescBuffer

 more3:
  mov al,[si]
  mov [di],al
  inc si
  inc di
  cmp al,"$"
  jne more3

 newline
 loop loopProd1
```

```
endm

inputForGeneral macro quoNumber
 mov ah,0Ah
       mov dx, offset quoNumber
       int 21h

       mov si,offset quoNumber + 1
       mov cl,[si]
       mov ch,0
       inc cx
       add si,cx
       mov al,'$'
       mov [si],al
    newline
endm

outputForGeneral macro quoNumber
    mov ah,9
       mov dx,offset quoNumber +2
       int 21h

   newline
endm

tenderPrint macro
        newline
        print prodArrayOutputString
        newline
        print prodArrayTitleString
        newline
```

```
mov si, offset qtyArray

mov cl,numberOfProds

loop4:
;print srno
mov al,srno
add al,48
mov dl,al
mov ah,02h
int 21h

inc srno
space
space
space
space
space
space
;prod name
mov ah,9
mov dx, offset prodName
add dx,prodNameOutputIndex

int 21h

add prodNameOutputIndex,10

space


;prod desc
mov ah,9
```

```
mov dx,offset prodDesc
add dx,prodDescOutputIndex

int 21h
add prodDescOutputIndex,25
space


;print qtyArrayDisplayForOutput

mov dl, [si]
mov ah,02h
int 21h

mov dl,[si]
sub dl,48
add totalQty,dl

space
space
space
space
space



newline

inc si
loop loop4

newline
;total qty
```

```
        print totalQtyDisplay

        mov al,totalQty
        add al,48
        mov dl, al
        mov ah,02h
        int 21h

        mov srno,1
        mov prodNameOutputIndex,2
        mov prodDescOutputIndex,2
endm



poPrint macro
        newline
        print prodArrayOutputString
        newline
        print prodArrayTitleString
        newline

        mov si, offset qtyArray
        mov di, offset unitPriceArray

        mov cl,numberOfProds

        loop2:
        ;print srno
        mov al,srno
        add al,48
        mov dl,al
        mov ah,02h
        int 21h
```

```
inc srno
space
space
space
space
space
space
;prod name
mov ah,9
mov dx, offset prodName
add dx,prodNameOutputIndex

int 21h

add prodNameOutputIndex,10

space


;prod desc
mov ah,9
mov dx,offset prodDesc
add dx,prodDescOutputIndex

int 21h
add prodDescOutputIndex,25
space


;print qtyArrayDisplayForOutput

mov dl, [si]
mov ah,02h
```

```
int 21h

space
space
space
space
space
;print unitPriceArrayDisplayForOutput
mov dl,[di]
mov ah,02h
int 21h

space
space
space
space
space
space
space
space
space

; print amount
mov al,[si]
sub al,48
mov dl,[di]
sub dl,48
mul dl

add subtotal,al

add al,48
mov dl,al
mov ah,02h
```

```
int 21h



newline

inc si
inc di
loop loop2

newline
;subtotal amount
print subtotalDisplay

mov al,subtotal
add al,48
mov dl, al
mov ah,02h
int 21h

newline
;total qty
print totalQtyDisplay

mov al,totalQty
add al,48
mov dl, al
mov ah,02h
int 21h



endm
```

```
.MODEL LARGE
.STACK 1000H

DATA SEGMENT
M1 DB 10,13,10,13,'              *  Welcome  *$',10,13
M2 DB 10,13,10,13,'Enter your Choise $'

M3 DB 10,13,'  *        1.Purchase Order                *$'
M4 DB 10,13,'  *        2.Tender Purchase Order Binding         *$'



;STAR RESIZE


MR1 DB 10,13,'  *                               *$'
MR2 DB 10,13,'  ***$'

MR3 DB 10,13,'  *                               *$'


MR4 DB 10,13,'  *                                               *$'
MR5 DB 10,13,'  ***$'



MR6 DB 10,13,'  *                               *$'
MR7 DB 10,13,'  **$'
```

```
SEJ DB 10,13,10,13,' $'


    how_many_items DB 0DH, 0AH, "How many Items?:" ,0DH, 0AH, "$"
    how_many DB 0DH, 0AH, "Please specify quantity for item:" ,0DH, 0AH, "$"
    price_dis db 0DH, 0AH, "Please enter price:" ,0DH, 0AH, "$"
    newLinez db 0DH, 0AH, "$"


    dottedline DB 0DH, 0AH, "---------------------------------------" ,0DH, 0AH, "$"
    itemheader DB 0DH, 0AH, "ITEM | ITEM QUANT. | ITEM PRICE | Tot" ,0DH, 0AH, "$"
    blankspace db 0DH,0AH, "      " ,0DH, 0AH, "$"

    items DB 0DH, 0AH, "ITEM" ,0DH, 0AH, "$"
    totalis DB 0DH, 0AH, "TOTAL AMOUNT: " ,0DH, 0AH, "$"
    quan DW 09H DUP (?)
    price DW 09H DUP (?)
    correstotal DW 09H DUP(?)
    totals DW 00H
    counter DB 00H
    aux DB 00H
    totalcount DW 00H


    numberOfProds db 0
srno db 1
index dw 0
printindex dw 0
taxPercent db 0
subtotal db 0
total db 0
discount db 0
```

```
totalQty db 0
totalQtyDisplay db 'Total Qty:$'

inputIndexProdName dw 0
inputIndexProdDesc dw 0

prodNameIndex dw 0
prodNameOutputIndex dw 2
prodDescOutputIndex dw 2
;integer array
qtyArray db 9 dup(?)
unitPriceArray db 9 dup(?)
amountArray db 9 dup(?)
;string array
prodName dw 9*10 dup(0)
prodNameInputDisplay dw 'Prod Name:$'
prodNameBuffer dw 10
        dw ?
        dw 10 dup(0)

prodDesc dw 9*25 dup(0)
prodDescInputDisplay dw 'Prod Desc:$'
prodDescBuffer dw 25
        dw ?
        dw 25 dup(0)



prodArrayDisplayString db 'Enter Product details for each product :$'
prodArrayTitleString db 'SR.NO.  Name  Desc    Qty  Unit Price Amount$'

numberOfProdsDisplayForInput db 'Enter number of products :$'
qtyArrayDisplayForInput db 'Quantity:$'
unitPriceArrayDisplayForInput db 'Unit price:$'
```

```
qtyArrayDisplayForOutput db 'Qty:$'
unitPriceArrayDisplayForOutput db 10,'Unit Price:Rs.'

prodArrayOutputString db 'Prod Array :$'
totalDisplay db 'Total Amount:$'
subtotalDisplay db 'Subtotal Amount:$'
discountPercentInputDisplay db 'Enter Discount percent:$'



; general inputs for tender , po , buying contract
tenderInputMsg db '--TENDER: Request for quotation--$'
unitPricesInputMsg db 'Enter Unit Prices from supplier:$'



;Tender : request for quotation
quoNumber db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user

quoNumberInputDisplay db 'Enter quotation number:$'
quoNumberOutputDisplay db 'Quotation number:$'

vendorName db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
vendorNameInputDisplay db 'Enter vendor name:$'
vendorNameOutputDisplay db 'Vendor name:$'

vendorRefId db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
vendorRefIdInputDisplay db 'Enter vendor RefID:$'
```

```
vendorRefIdOutputDisplay db 'Vendor RefID:$'

buyerCompanyName db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
buyerCompanyNameInputDisplay db 'Enter buyer Company Name:$'
buyerCompanyNameOutputDisplay db 'Buyer company:$'

dateOfTender db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
dateOfTenderInputDisplay db 'Enter Date of Tender:$'
dateOfTenderOutputDisplay db 'Date of Tender:$'


; PO : generals
poMessage db '--PURCHASE ORDER--$'

poNumber db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
poNumberInputDisplay db 'Enter PO Number:$'
poNumberOutputDisplay db 'PO Number:$'


poDate db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
poDateInputDisplay db 'Enter PO Date:$'
poDateOutputDisplay db 'PO Date:$'


poEDate db 26       ; max characters allowed
        db ?        ; number of characters entered by user
```

```
        db 26 dup(0) ; characters entered by user
poEDateInputDisplay db 'Enter Expected Delivery Date:$'
poEDateOutputDisplay db 'Expected Delivery Date:$'

poBuyerPhone db 26      ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
poBuyerPhoneInputDisplay db 'Enter Buyer Phone number:$'
poBuyerPhoneOutputDisplay db 'Buyer Phone Number :$'


poBuyerEmail db 26      ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
poBuyerEmailInputDisplay db 'Enter Buyer Email:$'
poBuyerEmailOutputDisplay db 'Buyer Email :$'


poBuyerAddr db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
poBuyerAddrInputDisplay db 'Enter Buyer Address:$'
poBuyerAddrOutputDisplay db 'Buyer Address :$'


poVendorName db 26      ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
poVendorNameInputDisplay db 'Enter Vendor Name:$'
poVendorNameOutputDisplay db 'Vendor Name :$'

poVendorAddr db 26      ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
```

```
poVendorAddrInputDisplay db 'Enter Vendor Address:$'
poVendorAddrOutputDisplay db 'Vendor Address :$'

poBuyerShipToAddr db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
poBuyerShipToAddrInputDisplay db 'Enter Buyer Ship To Address:$'
poBuyerShipToAddrOutputDisplay db 'Buyer Ship To Address :$'

poBuyerRequistioner db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
poBuyerRequistionerInputDisplay db 'Enter Buyer Requistioner:$'
poBuyerRequistionerOutputDisplay db 'Buyer Requistioner:$'

poBuyerShipVia db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
poBuyerShipViaInputDisplay db 'Enter Ship to Buyer Via:$'
poBuyerShipViaOutputDisplay db 'Ship to buyer via:$'

poBuyerShipTerms db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
poBuyerShipTermsInputDisplay db 'Enter Shipping terms:$'
poBuyerShipTermsOutputDisplay db 'Shipping Terms:$'

bindingItemNos db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
bindingItemNosInputDisplay db 'Enter Urgent product nos(comma separated):$'
bindingItemNosOutputDisplay db 'Urgent products:$'

bindingAdvance db 26       ; max characters allowed
```

```
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
bindingAdvanceInputDisplay db 'Enter Advance payment percent:$'
bindingAdvanceOutputDisplay db 'Advance Payment percent:$'


bindingAfterShip db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
bindingAfterShipInputDisplay db 'Enter After Shipment payment percent:$'
bindingAfterShipOutputDisplay db 'After Shipment Payment percent:$'


bindingAfterDelivery db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
bindingAfterDeliveryInputDisplay db 'Enter After Delivery payment percent:$'
bindingAfterDeliveryOutputDisplay db 'After Delivery Payment percent:$'

bindingDeliveryState db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
bindingDeliveryStateInputDisplay db 'Enter Delivery State:$'
bindingDeliveryStateOutputDisplay db 'Delivery State:$'


bindingInvoiceState db 26       ; max characters allowed
        db ?        ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
bindingInvoiceStateInputDisplay db 'Enter Invoice State:$'
bindingInvoiceStateOutputDisplay db 'Invoice State:$'

bindingAnnual db 26       ; max characters allowed
        db ?        ; number of characters entered by user
```

```
        db 26 dup(0) ; characters entered by user
bindingAnnualInputDisplay db 'Enter Annual requirement:$'
bindingAnnualOutputDisplay db 'Annual requirement:$'


bindingMonthly db 26       ; max characters allowed
        db ?       ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
bindingMonthlyInputDisplay db 'Enter Monthly requirement:$'
bindingMonthlyOutputDisplay db 'Monthly requirement:$'


bindingQuarterly db 26       ; max characters allowed
        db ?       ; number of characters entered by user
        db 26 dup(0) ; characters entered by user
bindingQuarterlyInputDisplay db 'Enter Quarterly requirement:$'
bindingQuarterlyOutputDisplay db 'Quarterly requirement:$'



; Binding contract cases : taking inputs based on case selected
caseMsgDisplay db 'Choose type of binding contract for sending PO to supplier:$'
caseMsg1 db 'Enter 1 for normal circumstances$'
caseMsg2 db 'Enter 2 for informing supplier over the phone$'
caseMsg3 db 'Enter 3 for requesting supplier to make early delivery$'
caseMsg4 db 'Enter 4 for requesting supplier to do partial delivery$'
caseMsg5 db 'Enter 5 for asking for proforma invoice for advance payment$'
caseMsg6 db 'Enter 6 for difference in billing and delivery address$'
caseMsg7 db 'Enter 7 for delivery of goods monthly$'
caseMsg8 db 'Enter 8 for delivery of goods quarterly$'
caseMsgInputDisplay db 'Enter choice:$'
errorMsg db 'Invalid choice$'
```

caseMsg db ?

case1Output dw 'Buyer Contract:Dear Supplier, Greetings. Please find attached PO poNumber against submitted quotation quoNumber . Kindly confirm receipt of the same and send order acknowledgment by return email. With best regards buyerCompanyName .$'

case2Output dw 'Buyer Contract:Dear Supplier , Greetings. Please find attached PO poNumber . Also kindly make arrangements to submit the Order Acknowledgement at the earliest. Also, as discussed we need delivery of the items by expectedDateOfDelivery. Please expedite the order accordingly.With best regards buyerCompanyName .$'

case3Output dw 'Buyer Contract:Dear Supplier. Please find attached PO poNumber against the submitted quotation quoNumber . Please send acknowledgement for the same. As per the quotation the delivery will be in 4 weeks from the date of receipt of PO . However as we have explained earlier that we require the requested items on an immediate basis. We will appreciate it if you can make arrangements for the delivery within the next 2 weeks. For any assistance , please feel free to reach us.Thanking you , Best Regards buyerCompanyName.$'

case4Output dw 'Buyer Contract:Dear Supplier, Please find attached PO poNumber . Kindly confirm receipt of the same and send acknowledgement by return email. Also, as explained over the phone the inventory fo reordered item numbers itemNos is very less. We request to please make arrangements to deliver these items within the next 2 weeks . Rest of the items can be dispatched as per delivery schedule.Thanking you , Best Regards buyerCompanyName.$'

case5Output dw 'Buyer Contract:Dear Supplier, Please find attached PO poNumber. Kindly confirm receipt of the same and send acknowledgement by return email. As per agreed payment terms its __advanceP% advance , __afterShipP% against shipping documents and remaining __afterDeliveryP% after receipt of the complete order. Please send the Proforma Invoice for the __advanceP% advance payment so that we can make arrangements to process the same. Thank you , Best Regards __buyerCompanyName.$'

case6Output dw 'Buyer Contract:Dear Supplier , Greetings. Please find attached PO poNumber .Kindly acknowledge receipt of the same. Also, please follow below given instructions regarding the delivery. The order to be Delivery address located in deliveryState and all the invoices to be sent to our invoiceState. Addresses are given in detail in the PO.$'

case7Output dw 'Buyer Contract:Dear Supplier . Attached is the PO poNumber against the submitted quotation __quoNumber. As discussed , we are releasing the order for __annualRequirement pcs which is our annual requirement.However, we need monthly delivery

of monthlyRequiremnt pcs . You can refer to the same PO on the invoices with the following comment : PO _poNumber _ (1st Lot) , (2nd Lot) …(12th Lot).$'

case8Output dw 'Buyer Contract: Dear Supplier . Attached is the PO poNumber against the submitted quotation __quoNumber. As discussed , we are releasing the order for __annualRequirement pcs which is our annual requirement.However, we need monthly delivery of quarterlyRequiremnt pcs . You can refer to the same PO on the invoices with the following comment : PO _poNumber _ (1st Lot) , (2nd Lot),(3rd Lot),(4th Lot).$'

DATA ENDS

CODE SEGMENT

START:

MOV AX,@DATA
MOV DS,AX

TOP:

LEA DX,M1
MOV AH,9
INT 21H

LEA DX,SEJ ;NEWLINE
MOV AH,9
INT 21H

LEA DX,MR2
MOV AH,9

```
INT 21H

LEA DX,MR2
MOV AH,9
INT 21H     ;BORDER

LEA DX,MR3
MOV AH,9
INT 21H

LEA DX,MR3
MOV AH,9
INT 21H




LEA DX,M3
MOV AH,9
INT 21H



LEA DX,M4
MOV AH,9
INT 21H


LEA DX,MR1
MOV AH,9
INT 21H

LEA DX,MR1
MOV AH,9
```

```
INT 21H

LEA DX,MR2
MOV AH,9
INT 21H

LEA DX,MR2
MOV AH,9
INT 21H

LEA DX,M2
MOV AH,9
INT 21H



MOV AH,1
INT 21H
MOV BH,AL
SUB BH,48

MOV CH,BH

CMP BH,1
JE ad1

CMP CH,2
JE ad3

ad1:

  mov dx, offset how_many_items
   mov ah, 09H
   int 21h
```

```
MOV AH, 00H


mov dx,offset newLinez
mov ah,09H
int 21h

mov ah,01
int 21h

mov bl,al
mov ax,0000H
mov al,bl
SUB al,30H

mov CX,AX
mov totalcount,CX

MOV DI, OFFSET price
MOV SI, OFFSET quan



here:

mov dx,offset how_many
mov ah,09H
int 21h

MOV ah,00H

mov ah,01H
int 21h
```

```
SUB al,30H

mov [SI],al
ADD SI,0002H

mov dx,offset newLinez
mov ah,9
int 21h


mov dx,offset price_dis
mov ah,9
int 21h


mov ah,01
int 21h

SUB al,30H

MOV BL,10H
MUL BL


MOV BL,AL

MOV AL,00H
mov ah,00H

mov ah,01H
int 21h

SUB al,30H
```

```
ADD AL,BL



mov [DI],al
ADD DI,0002H

mov dx,offset newLinez
mov ah,9
int 21h

loop here

MOV bx, offset correstotal
MOV DI, OFFSET price
MOV SI, OFFSET quan

MOV CX,0009H

corres:

MOV AL,[DI]
MUL [SI]

MOV [BX],al

ADD SI,0002H
ADD DI,0002H
ADD BX,0002H

loop corres
```

```
mov dx,offset dottedline
mov ah,09H
int 21h

mov dx,offset newLinez
mov ah,09H
int 21h

mov dx,offset itemheader
mov ah,09H
int 21h

mov dx,offset newLinez
mov ah,09H
int 21h

mov dx,offset dottedline
mov ah,09H
int 21h


mov dx,offset newLinez
mov ah,09H
int 21h


MOV BX, offset correstotal
MOV DI, OFFSET price
MOV SI, OFFSET quan
```

```
MOV al,1H
MOV CX,totalcount

billgenerate:

mov dx, offset items
mov ah, 09H
int 21h

mov dl,counter
add dl,48
mov ah,02
int 21H

mov al,counter
inc al
mov counter,al

mov dl,00h

  mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
```

```
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

  mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H
```

```
mov dl,[SI]


 add dl,48
mov ah,02
int 21H

ADD SI,0002H

mov dl,00h

mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h
```

```
MOV DX,0000H

  mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

  mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
```

```
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H


MOV al,[DI]
AND al,0F0H

shr al,1
shr al,1
shr al,1
shr al,1


mov dl,al
add dl,48
mov ah,02
int 21H

mov al,[DI]
AND al,00FH
```

```
mov dl,al
add dl,48
mov ah,02
int 21H

ADD DI,0002H

mov dl,00h

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
```

```
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H

 mov dl,32   ; blank space
mov ah,02H
int 21h

MOV DX,0000H


mov al,[bx]
AND al, 0F0H


shr al,1
shr al,1
shr al,1
shr al,1
```

```
mov dl,al
add dl,48
mov ah,02
int 21H

mov al,[bx]
AND al, 00FH


mov dl,al
add dl,48
mov ah,02
int 21H

ADD BX,0002H

mov dl,00h


mov dx,offset dottedline
mov ah,09H
int 21h
MOV DX,0000H


mov dx,offset newLinez
mov ah,09H
int 21h

MOV DX,0000H

loop billgenerate
```

```
mov dx,offset totalis
mov ah,09H
int 21h


mov SI, offset correstotal
MOV CX,0009H
MOV AL,00H

tot:

ADD AL,[SI]
ADD SI,0002H


loop tot


mov aux,al

mov bl,al
AND bl,0F0H

shr bl,1
shr bl,1
shr bl,1
shr bl,1


mov dl,bl
add dl,48
mov ah,02H
int 21h
```

```
 mov bl,aux
 AND bl,00FH

 mov dl,bl
 add dl,48
 mov ah,02H
 int 21h

 JMP TOP

ad3:

;quoNumber
    newline
     print quoNumberInputDisplay
     inputForGeneral quoNumber




     ;vendorName
     print vendorNameInputDisplay
     inputForGeneral vendorName



     ;vendorRefID
     print vendorRefIdInputDisplay
     inputForGeneral vendorRefId


     ;buyerCompanyName
     print buyerCompanyNameInputDisplay
     inputForGeneral buyerCompanyName
```

```
;dateOfTender
print dateOfTenderInputDisplay
inputForGeneral dateOfTender




; input numberofProds
print numberOfProdsDisplayForInput

mov ah,01h
int 21h
sub al,30h
mov ch,0h
mov numberOfProds,al
mov cl,numberOfProds


; input for prod Name , prod Desc

inputProdName

inputProdDesc

mov cl,numberOfProds
;mov cx,9
mov si,offset qtyArray
mov di,offset unitPriceArray

newline
;print 'Enter 9 values in array:'
```

```
print prodArrayDisplayString

newline

;input qty,prodNameArrayItem,prodDescArrayItem from user
loop1:


    print qtyArrayDisplayForInput
    mov ah,01h
    int 21h

    mov [si],al
    inc si

    space


    newline
    loop loop1

 ; input discountRate
 ;newline
; print discountPercentInputDisplay
; mov ah,01h
 ;int 21h
; mov discount,al


 ; TENDER DISPLAY
 ; display tender the user without unit price
  print tenderInputMsg
  newline
```

```
  print quoNumberOutputDisplay
  outputForGeneral quoNumber

  print vendorNameOutputDisplay
  outputForGeneral vendorName

  print vendorRefIdOutputDisplay
  outputForGeneral vendorRefId

  print buyerCompanyNameOutputDisplay
  outputForGeneral buyerCompanyName

  print dateOfTenderOutputDisplay
  outputForGeneral dateOfTender

  tenderPrint

;total amount
;mov al,discount
;sub al,48
;mov bl, subtotal

;mul bl

;mov bh,al
;mov al,100

;div bh

;add al,bl

;print totalDisplay
;add al,48
;mov dl,al
```

```
;mov ah,02h
;int 21h



; TAKE UNIT PRICE AS INPUT
  newline
  print unitPricesInputMsg
  mov cl,numberOfProds
  mov di,offset unitPriceArray
  newline
  loop3:
   print unitPriceArrayDisplayForInput
   mov ah,01h
   int 21h

   mov [di],al
   inc di

   newline
  loop loop3

; TAKE OTHER PO AS INPUT
newline
 print poNumberInputDisplay
 inputForGeneral poNumber

 print poDateInputDisplay
 inputForGeneral poDate

 print poEDateInputDisplay
 inputForGeneral poEDate

 print poBuyerPhoneInputDisplay
```

```
inputForGeneral poBuyerPhone

print poBuyerEmailInputDisplay
inputForGeneral poBuyerEmail

print poBuyerAddrInputDisplay
inputForGeneral poBuyerAddr

print poVendorNameInputDisplay
inputForGeneral poVendorName

print poVendorAddrInputDisplay
inputForGeneral poVendorAddr

print poBuyerShipToAddrInputDisplay
inputForGeneral poBuyerShipToAddr

print poBuyerRequistionerInputDisplay
inputForGeneral poBuyerRequistioner

print poBuyerShipViaInputDisplay
inputForGeneral poBuyerShipVia

print poBuyerShipTermsInputDisplay
inputForGeneral poBuyerShipTerms


;DISPLAY PO and TAKE INPUT FOR BINDING CONTRACT
newline

print poMessage
newline

 ;quoNumber
```

```
print quoNumberOutputDisplay
outputForGeneral quoNumber

;poNumber
print poNumberOutputDisplay
outputForGeneral poNumber

;poDate
print poDateOutputDisplay
outputForGeneral poDate

;poEDate
print poEDateOutputDisplay
outputForGeneral poEDate

;buyerCompnanyName
print buyerCompanyNameOutputDisplay
outputForGeneral buyerCompanyName

;buyer phone
print poBuyerPhoneOutputDisplay
outputForGeneral poBuyerPhone

;buyer email
print poBuyerEmailOutputDisplay
outputForGeneral poBuyerEmail

;buyer addr
print poBuyerAddrOutputDisplay
outputForGeneral poBuyerAddr

; vendor name
print poVendorNameOutputDisplay
outputForGeneral poVendorName
```

```
; vendor addr
 print poVendorAddrOutputDisplay
 outputForGeneral poVendorAddr

; buyer shipto addr
 print poBuyerShipToAddrOutputDisplay
 outputForGeneral poBuyerShipToAddr

 ; buyer requistioner
 print poBuyerRequistionerOutputDisplay
 outputForGeneral poBuyerRequistioner

 ; buyer ship via
 print poBuyerShipViaOutputDisplay
 outputForGeneral poBuyerShipVia

 ;poBuyerShipTerms
 print poBuyerShipTermsOutputDisplay
 outputForGeneral poBuyerShipTerms


newline

poPrint

newline

print caseMsgDisplay
newline
print caseMsg1
newline
print caseMsg2
newline
```

```
print caseMsg3
newline
print caseMsg4
newline
print caseMsg5
newline
print caseMsg6
newline
print caseMsg7
newline
print caseMsg8
newline

tryagain:
print caseMsgInputDisplay

mov ah,01h
int 21h
mov [caseMsg] , al
newline
cmp [caseMsg],31h
je @case1

cmp [caseMsg],32h
je @case2

cmp [caseMsg],33h
je @case3

cmp [caseMsg],34h
je @case4

cmp [caseMsg],35h
je @case5
```

```
cmp [caseMsg],36h
je @case6

cmp [caseMsg],37h
je @case7

cmp [caseMsg],38h
je @case8


jmp @default




; DISPLAY PO : other PO , prod table with unit price, amount
; DISPLAY BINDING CONTRACT : accroding to type of binding contract selected

@case1:
 print case1Output
 newline
 print poNumberOutputDisplay
 outputForGeneral poNumber

 newline
 print quoNumberOutputDisplay
 outputForGeneral quoNumber

 newline
 print buyerCompanyNameOutputDisplay
 outputForGeneral buyerCompanyName

 jmp @end
```

```
@case2:
 print case2Output
 newline
 print poNumberOutputDisplay
 outputForGeneral poNumber

 newline
 print buyerCompanyNameOutputDisplay
 outputForGeneral buyerCompanyName


jmp @end

@case3:
 print case3Output
 newline
 print poNumberOutputDisplay
 outputForGeneral poNumber

 newline
 print quoNumberOutputDisplay
 outputForGeneral quoNumber

 newline
 print buyerCompanyNameOutputDisplay
 outputForGeneral buyerCompanyName



 jmp @end

@case4:
 print bindingItemNosInputDisplay
```

```
inputForGeneral bindingItemNos


print case4Output
newline
print poNumberOutputDisplay
outputForGeneral poNumber

newline
print bindingItemNosOutputDisplay
outputForGeneral bindingItemNos

newline
print buyerCompanyNameOutputDisplay
outputForGeneral buyerCompanyName


jmp @end

@case5:
 print bindingAdvanceInputDisplay
 inputForGeneral bindingAdvance

 print bindingAfterShipInputDisplay
 inputForGeneral bindingAfterShip

 print bindingAfterDeliveryInputDisplay
 inputForGeneral bindingAfterDelivery


 print case5Output

newline
print poNumberOutputDisplay
```

```
outputForGeneral poNumber

newline
print bindingAdvanceOutputDisplay
outputForGeneral bindingAdvance

newline
print bindingAfterShipOutputDisplay
outputForGeneral bindingAfterShip

newline
print bindingAfterDeliveryOutputDisplay
outputForGeneral bindingAfterDelivery

newline
print buyerCompanyNameOutputDisplay
outputForGeneral buyerCompanyName


jmp @end

@case6:
print bindingDeliveryStateInputDisplay
inputForGeneral bindingDeliveryState

print bindingInvoiceStateInputDisplay
inputForGeneral bindingInvoiceState

print case6Output


newline
print poNumberOutputDisplay
outputForGeneral poNumber
```

```
newline
print bindingDeliveryStateOutputDisplay
outputForGeneral bindingDeliveryState

newline
print bindingInvoiceStateOutputDisplay
outputForGeneral bindingInvoiceState




jmp @end

@case7:
print bindingAnnualInputDisplay
inputForGeneral bindingAnnual

print bindingMonthlyInputDisplay
inputForGeneral bindingMonthly

print case7Output


newline
print poNumberOutputDisplay
outputForGeneral poNumber

newline
print quoNumberOutputDisplay
outputForGeneral quoNumber

newline
```

```
print bindingAnnualOutputDisplay
outputForGeneral bindingAnnual

newline
print bindingMonthlyOutputDisplay
outputForGeneral bindingMonthly


jmp @end

@case8:
print bindingAnnualInputDisplay
inputForGeneral bindingAnnual

print bindingQuarterlyInputDisplay
inputForGeneral bindingQuarterly

print case8Output


newline
print poNumberOutputDisplay
outputForGeneral poNumber

newline
print quoNumberOutputDisplay
outputForGeneral quoNumber

newline
print bindingAnnualOutputDisplay
outputForGeneral bindingAnnual

newline
print bindingQuarterlyOutputDisplay
```

```
        outputForGeneral bindingQuarterly


        jmp @end

        @default:
         print errorMsg
         jmp tryagain


      @end:
      mov ah,4ch
      int 21h


      JMP TOP


CODE ENDS
END START
```