INDIVIDUAL ASSIGNMENT

TECHNOLOGY PARK MALAYSIA

CT074-3-2

Computer System Low Level Techniques

APU2F2106CS

Module Code	:	Computer System Low Level Techniques
Intake Code	:	APU2F2106CS
Hand out Date	:	25th November 2021
Hand in Date	:	21st January 2022

Project

Student ID	**Student Name**
TP058344	ARSLAN KHURRAM

Table of Contents

1.0 Abstract....1
2.0 Introduction to Assembly Language....1
2.1 Advantages of Assembly Language....2
2.2 Disadvantages of Assembly Language....2
3.0 Research and Malware Analysis....3
3.1 High Frequency Trading – HFT....3
3.2 Real-Time Embedded Systems....3
3.3 Digital Signal Processers – DSP....4
3.4 Use of Low Level Languages in Cybersecurity and forensic fields....4
3.5 Malware Analysis....5
4.0 System Design using Flowchart....9
5.0 System Screenshot....19
6.0 Source Code....28
7.0 Limitation....44
8.0 Conclusion & Self – Reflection....44
9.0 References....45

1.0 Abstract

A scenario is assigned to me that I am an member of Asia Pacific University's Cyber Security Club and I have tasked to develop a Cashing Program in Assembly for an upcoming event. The System has the ability to display the menu to the customers from which they can choice various functions such as Workshops, Challenges, Activities, Donations, Remaining Seats, and the entire Price Structure. The sub-menu shows even more of its functionalities from which customer choose and their shopping cart will be updated in iterations. Lastly when Checkout select the program displays Total and Need to Pay depending on the membership of user will displayed on the screen.

2.0 Introduction to Assembly Language

High Level languages such as C, C++, Python, and Java have a good deal in common with natural languages (common English) that provide ease-of-life to programmers as the source code becomes easier to read and write. Whereas, the low-level languages machine code make it so that machines understand the source code better and that developers will develop better understanding of the machine and its architecture. To machines, high level languages is garbage and they can not make sense of the instructions. Only with the help of language translators known as compilers and interpreters the machine is able to convert each high-level instruction into many low level instructions so that it can make sense of the code. The complications do not end here as each of architecture such as Intel's microprocessor will understand machine code differently. So machine language is not shared in-between all machine and depending on microprocessors or mainframes the machine code will have different meaning.

The Low-level Machine language are nothing but bunch of zeros and ones only the most gift among us can understand it. As it hard to interpret it can be very tedious and error prone. Another low level language is Assemble language unlike machine code the source code does make sense for humans and it easier to understand. The relation between machine language and assembly language is that their instructions is one-to-one. Therefore, there are not many disadvantages of coding in assembly language. The greatest advantage of coding in assembly language is that in

terms of program execution it has better performance and saves much more memory relative to High Level Languages. As it was told before, high level language utilize compilers to convert it to Assembly language. The assembly language uses assemblers to convents it to Machine Language (Streib, 2020).

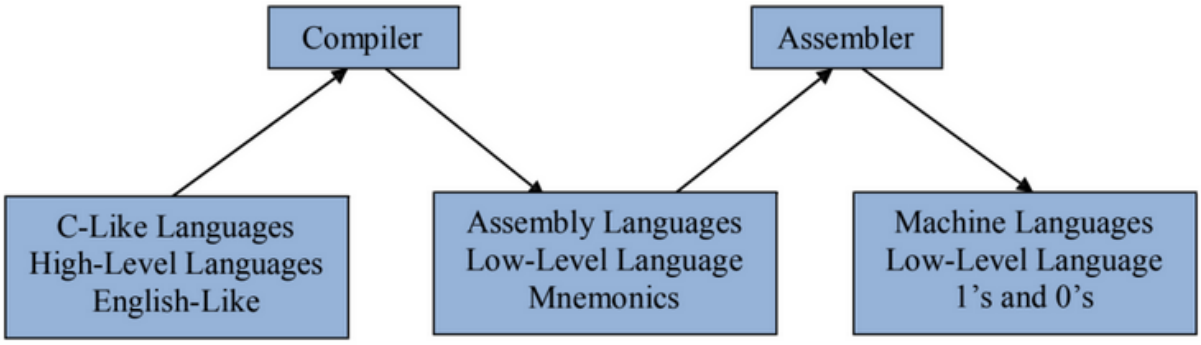


Fig 1 (Streib, 2020)

Presently the most prominent assemblers of x86 CPU architecture being utilized are Borland Turbo Assembler (TASM), Microsoft Macro Assembler (MASM) which are DOS/Windows-based. Lastly, Netwide Assember (NASM) which used every modern platform and supports many old platforms (*NASM*, n.d.) (HORSTR, 2020).

2.1 Advantages of Assembly Language

- Its highly memory efficient, and requires very little resources which allows it be perform will in budget or specialized hardware (Pedamkar, 2021).
- Performance of instruction being executed is very fast relative to high level languages (Streib, 2020).

2.2 Disadvantages of Assembly Language

- Development of the code takes highly skilled individual and a lot of time (Pedamkar, 2021).
- As the syntax and code is much closer to machines than humans, hence it is very difficult to make sense of it (Streib, 2020).
- The assembly code needs to be changed when porting it to another computer architecture (Streib, 2020).

3.0 Research and Malware Analysis

As assembly language has the numerous advantages in its arsenal, hence it has many real world applications. Typically theses applications are device drivers, Kernals, BIOS, real-time systems, low level embedded systems, boot loaders, emulators, and compilers (Naikis, 2015) (Ghoshal, 2016).

3.1 High Frequency Trading – HFT

The majority of financial enterprises that deal in stocks and other types of trading make use of High Frequency Trading (HFT) systems. These specific purpose systems are made using assembly language due its high performance, accuracy in transactions, and low latency. As time is very paramount for the traders even a few milliseconds of delay when purchasing stocks can be fatal. By implementations of these systems the enterprise can overtake the competitors in profit. As told before time is essence for the traders they consider waiting for high level language to be converted to machine code a big bottleneck. Consequently, research studies have proven the the companies that utilize High Frequency Trading Platforms saw a increase in total net income (Fernando, 2020).

3.2 Real-Time Embedded Systems

The creation of embedded systems can designed with the capabilities of real time computing depending on the circumstances. In these types of systems, time is really constrained and one-third of second can decide the difference life and death. Example of these systems implemented in society are air bags in cars as a vehicle comes to an abrupt stop encase of accident the air bag must come out in split second to save the passengers from fatal injury. Therefore, the embedded system implement in airbags must really high polling rate and detect and react to collisions instantaneously. Therefore, due these constraints the Real-time Operating Systems (RTOS) must be coded in assembly language to react quickly enough to a situation. Other examples of Real-time embedded systems are monitor inside the pacemaker, and robots at an assembly line (An Introduction to Real-Time Embedded Systems, 2019).

3.3 Digital Signal Processers – DSP

Digital Signal Processors are sensors that take analog signals or real-world signals from the environment. These analog signals can be voice, audio, video, temperature, pressure, gyro, or motion. These analog signals then need to be processed so that computer can understand them, hence they are converted to 1's and 0's by Analog-to-Digital converter. Depending on the requirements the rate at which these sensor need to process data has to be very fast. Programmer that code this process in assembly language remove bottlenecks increase performance of entire system. The implementation of assembly results in quick execution speed and it cuts down in manufacturing cost, therefore a company will outperform its competitors gain huge profits (Maxfield, 2007) (*A Beginner's Guide to Digital Signal Processing, n.d.).*

3.4 Use of Low Level Languages in Cybersecurity and forensic fields

Low Level Languages has direct control of computer's resources and they take a vital part in Cybersecurity and digital forensics. Source code that is written in low languages are hard for normal programmer to understand and in an off chance a criminal is able to get a hold of the source code considerable amount skills are required. Even most of hashing and encryption algorithm libraries are developed using low level languages (Heynthan et al., 2016).

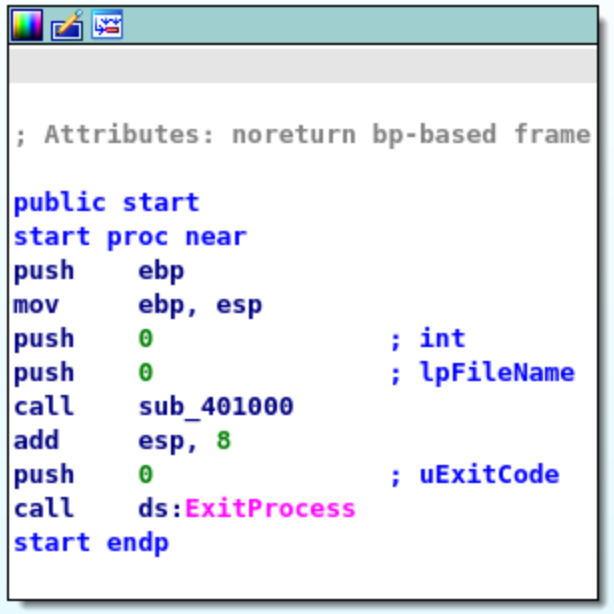
Another other way low level languages are being used is in reverse engineering in which a Cybersecurity expert converts code into a low level language. The way reverse engineering works is takes apart code in pieces and put it together in a low level language.

3.5 Malware Analysis

Malware Analysis is another sub field of Cybersecurity and Digital Forensics it concentrates how a malware effects a system and explain how the problem can reversed (Heynthan et al., 2016).

Ransomware1.exe

I have chosen ‘ransomware.exe’ for reverse engineering. It is an executable file and when it is executed by a user the entire file system user gets encrypted even the executable file.



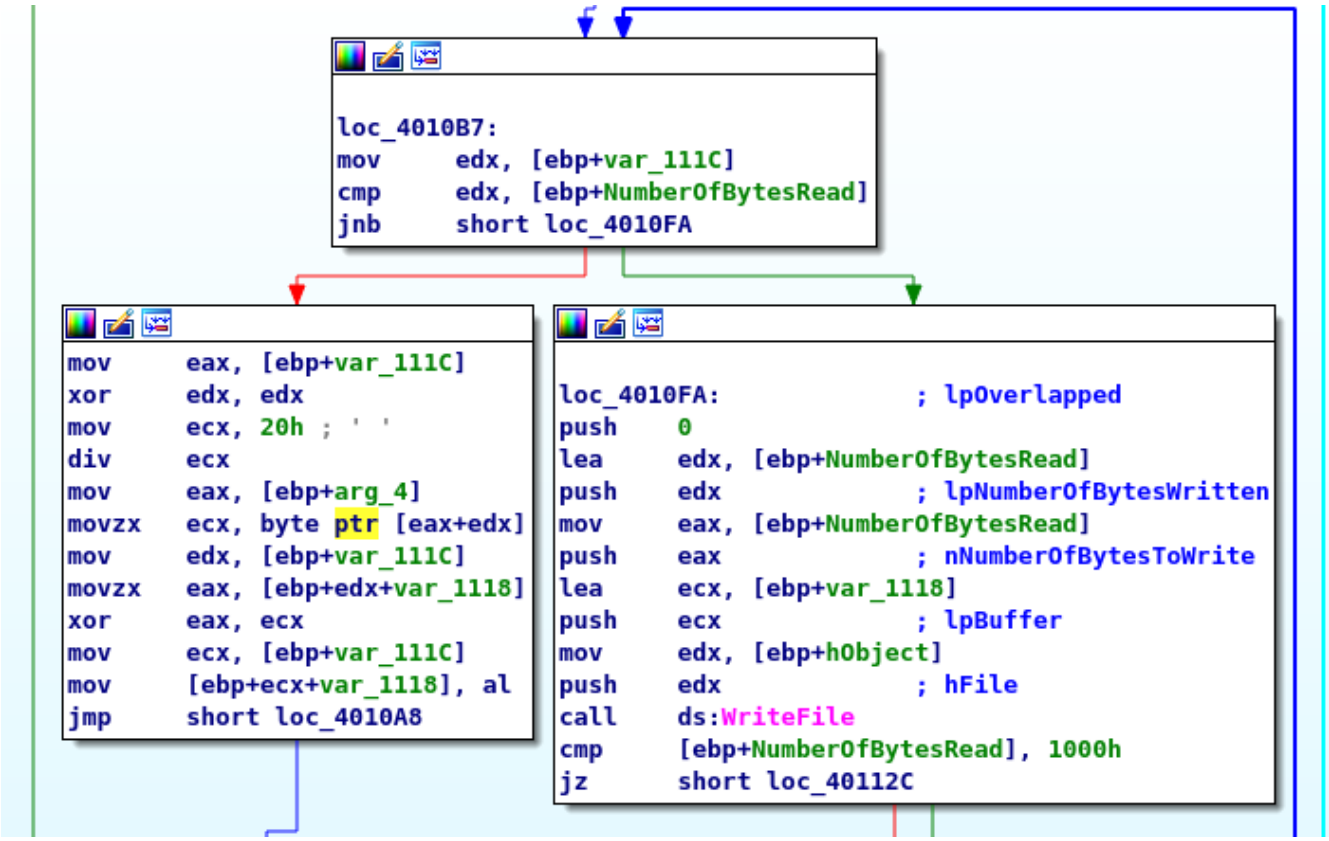
```
; Attributes: noreturn bp-based frame

public start
start proc near
push    ebp
mov     ebp, esp
push    0               ; int
push    0               ; lpFileName
call    sub_401000
add     esp, 8
push    0               ; uExitCode
call    ds:ExitProcess
start endp
```

This is the start function the used to call the function ‘sub_40100’ and push zero’s in a stack.

```
push    eax
push    offset Format   ; "%s_encrypted"
push    104h            ; BufferCount
lea     ecx, [ebp+Buffer]
push    ecx             ; Buffer
call    _snprintf
add     esp, 10h
push    0               ; hTemplateFile
push    80h             ; dwFlagsAndAttributes
push    3               ; dwCreationDisposition
push    0               ; lpSecurityAttributes
push    0               ; dwShareMode
push    80000000h       ; dwDesiredAccess
mov     edx, [ebp+lpFileName]
push    edx             ; lpFileName
call    ds:CreateFileA
mov     [ebp+hFile], eax
push    0               ; hTemplateFile
push    80h             ; dwFlagsAndAttributes
push    2               ; dwCreationDisposition
push    0               ; lpSecurityAttributes
push    0               ; dwShareMode
push    40000000h       ; dwDesiredAccess
lea     eax, [ebp+Buffer]
push    eax             ; lpFileName
call    ds:CreateFileA
mov     [ebp+hObject], eax
```

As shown above, new file it getting created and on each iteration of the file offset string '_encrypted' gets attached to name of the file.

At this stage the contents of file are getting read and check whether they exist or not.



As seen in of the image above, this is the core function of the entire program and this how the data gets encrypted. This section of the code keeps on looping over the contents in the file by checking and that the value in the register is zero or not. The block of code that is in the lower left

side of the image shows us few things about the encryption process. It tells us that the key is only 32 bytes long because of division. It also tells that entire encryption is processed via XOR GATE. Lastly, as told before the contents get looped around only one byte at a time. The ransomware has a simple encryption process, the var_111C is a byte_counter it counts the bytes inside a file. The remainder from the div is used get the position of the byte that needs to be encrypted. Look at the example below →

byte_count = 60

key = 32

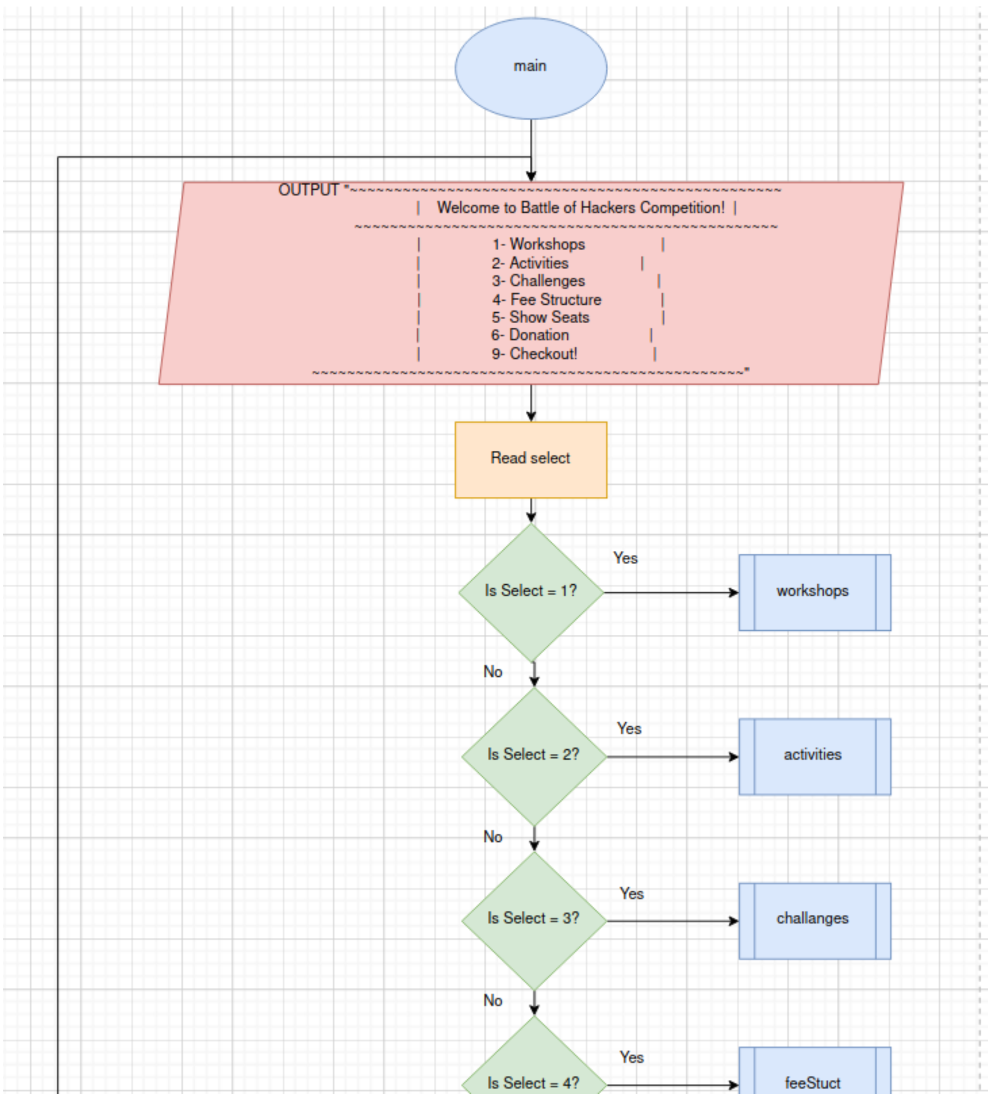
60 / 32 = Reminder - 28

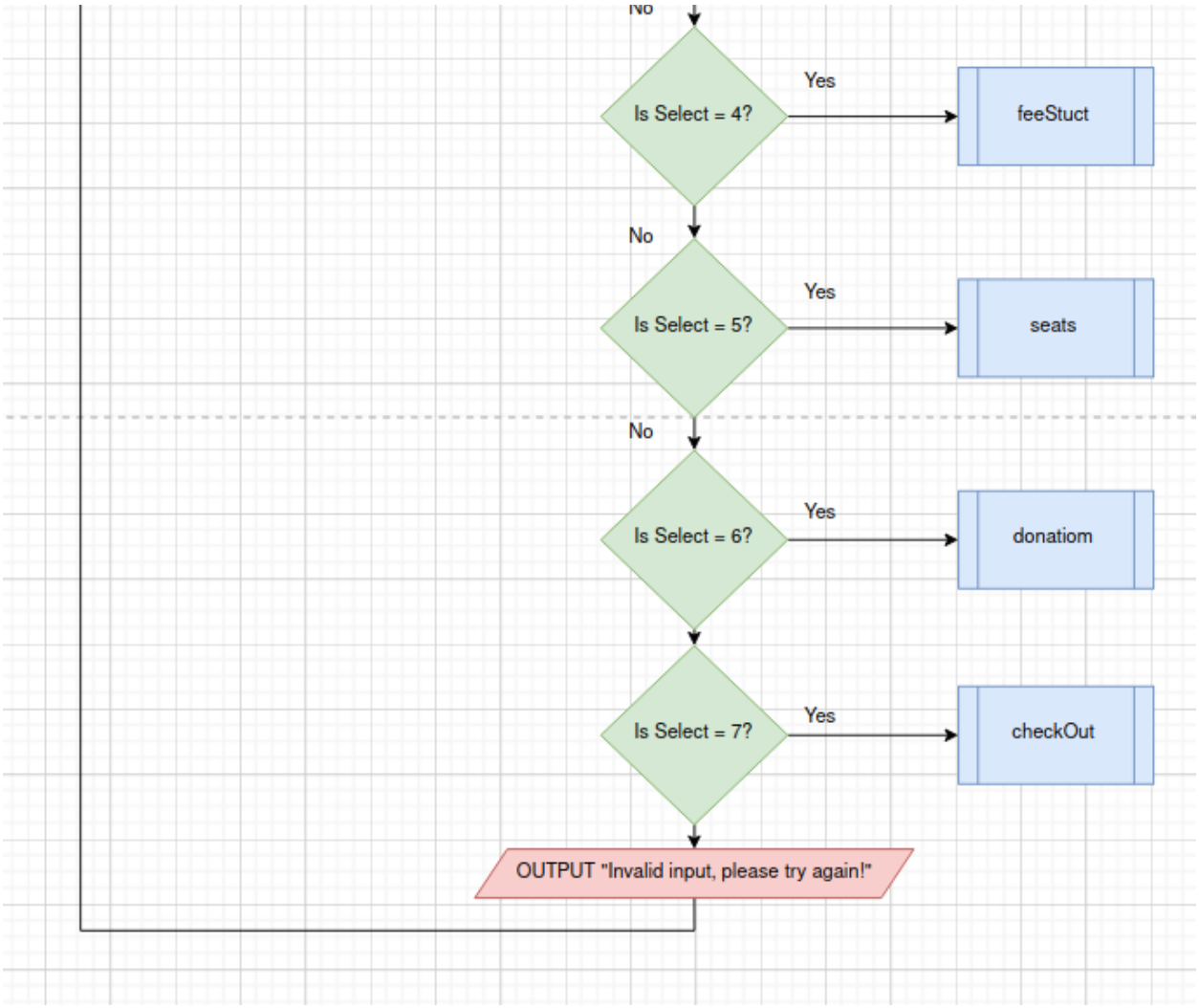
Therefore, index 120 will XORED with 28th byte of the key

To decrypt the database all we need to do is use the encrypted file and Xor the first 32 bytes of the file and then we will be able get our hands on key that was used to encrypt the files. With that key we will be able to decrypt our database.

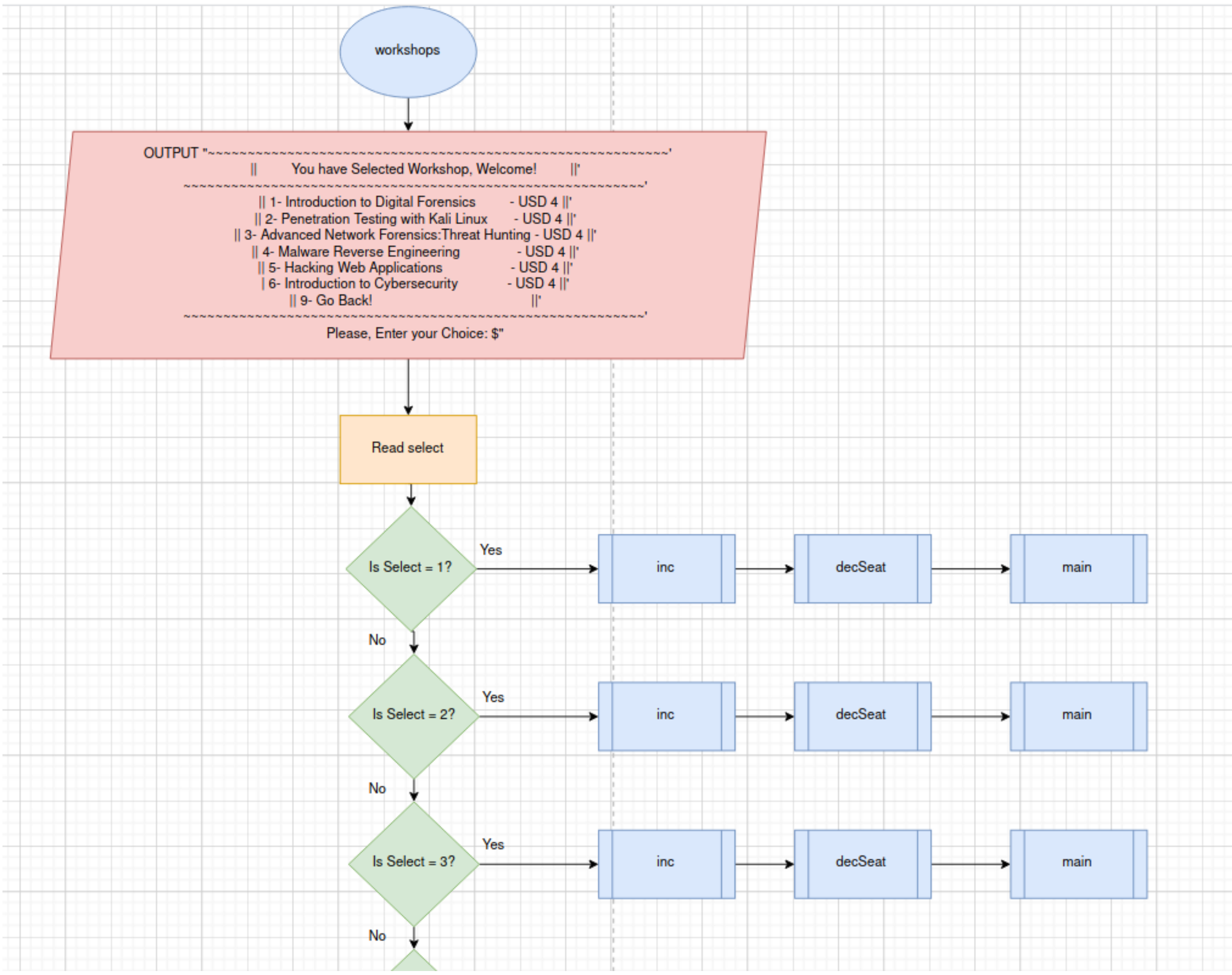
4.0 System Design using Flowchart

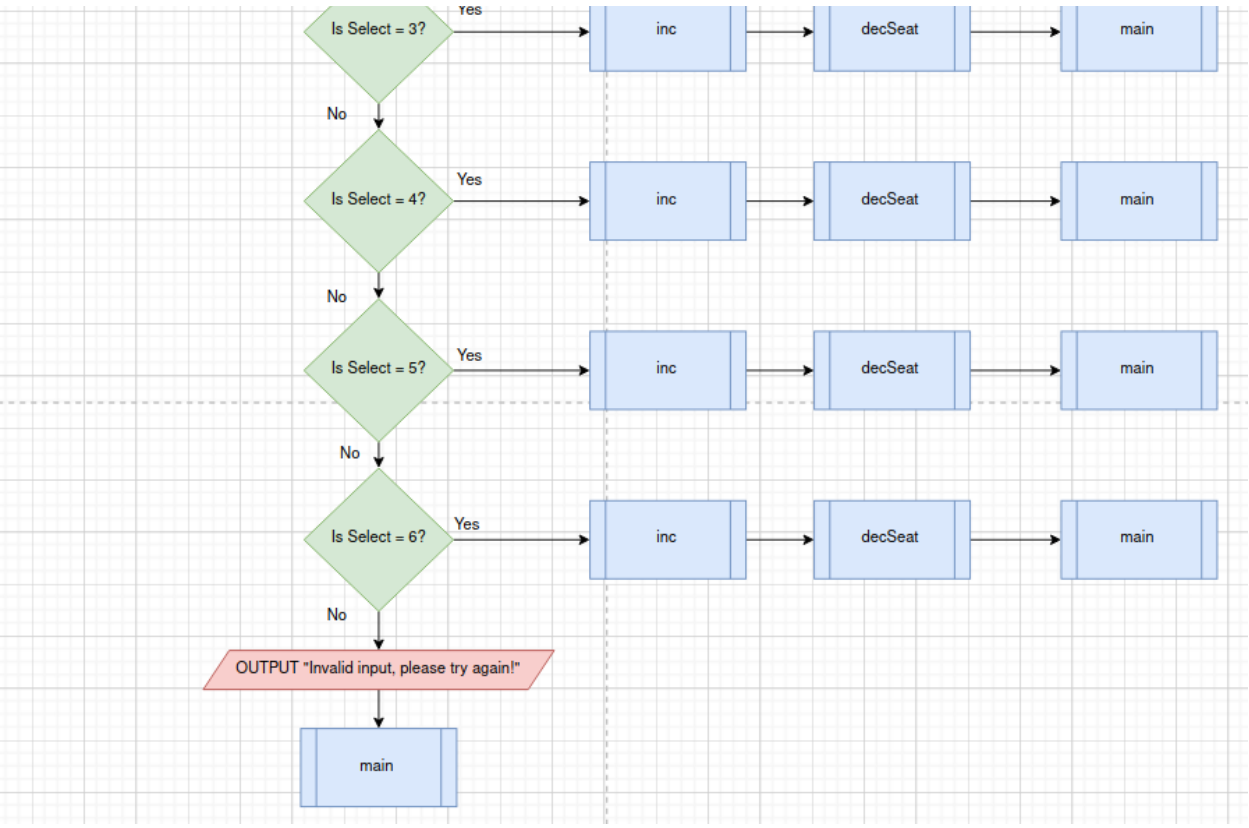
4.1 Welcome Menu



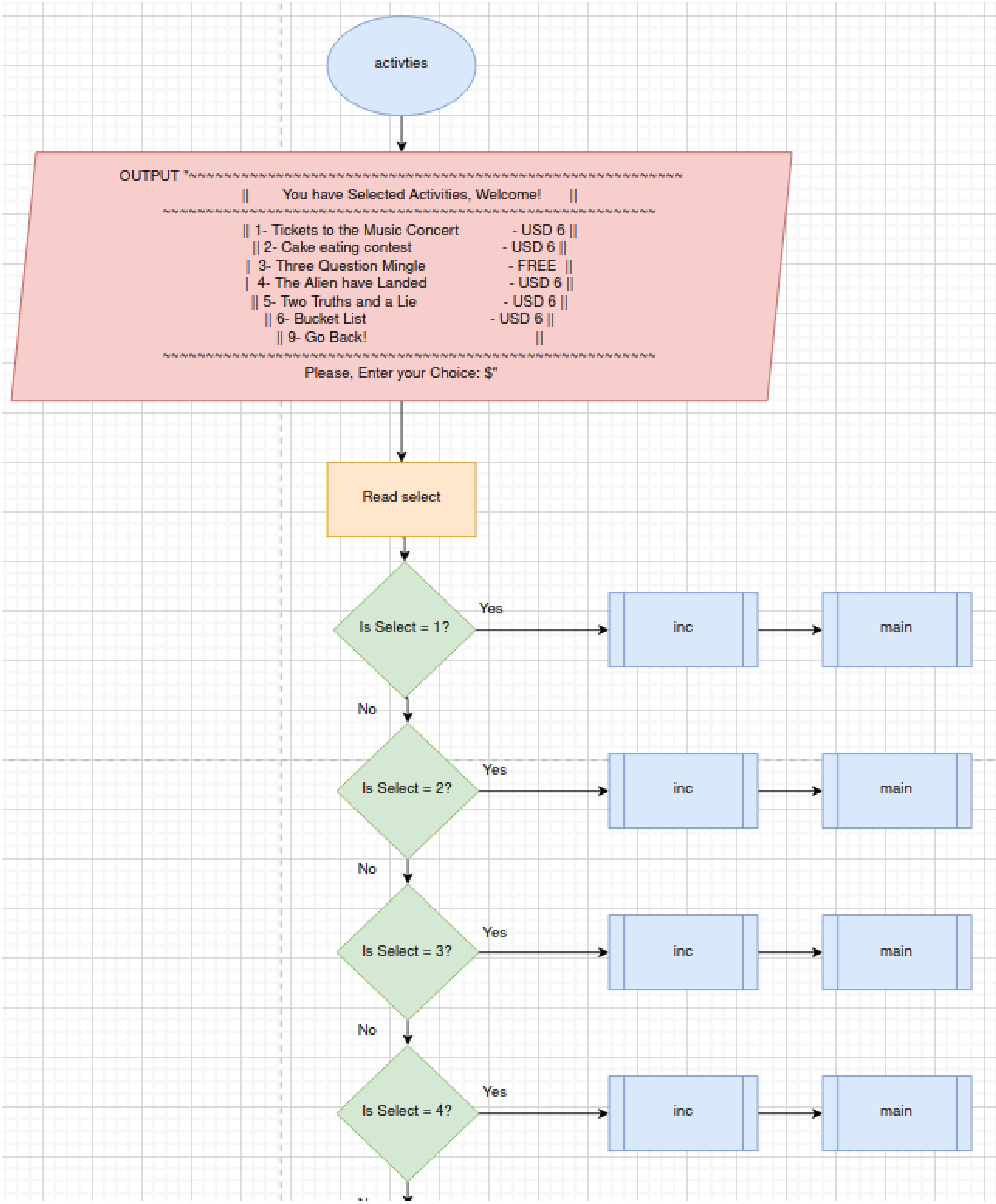

No
Is Select = 4?
Yes
feeStuct
No
Is Select = 5?
Yes
seats
No
Is Select = 6?
Yes
donatiom
Is Select = 7?
Yes
checkOut
OUTPUT "Invalid input, please try again!"

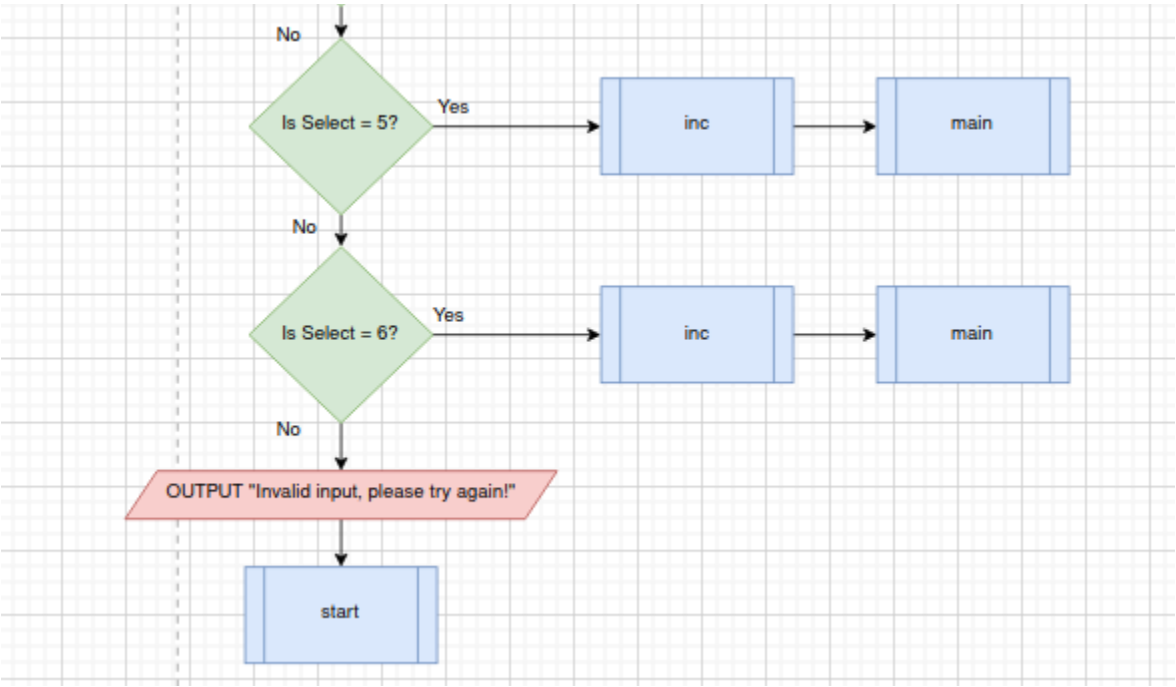
4.2 Workshop Menu



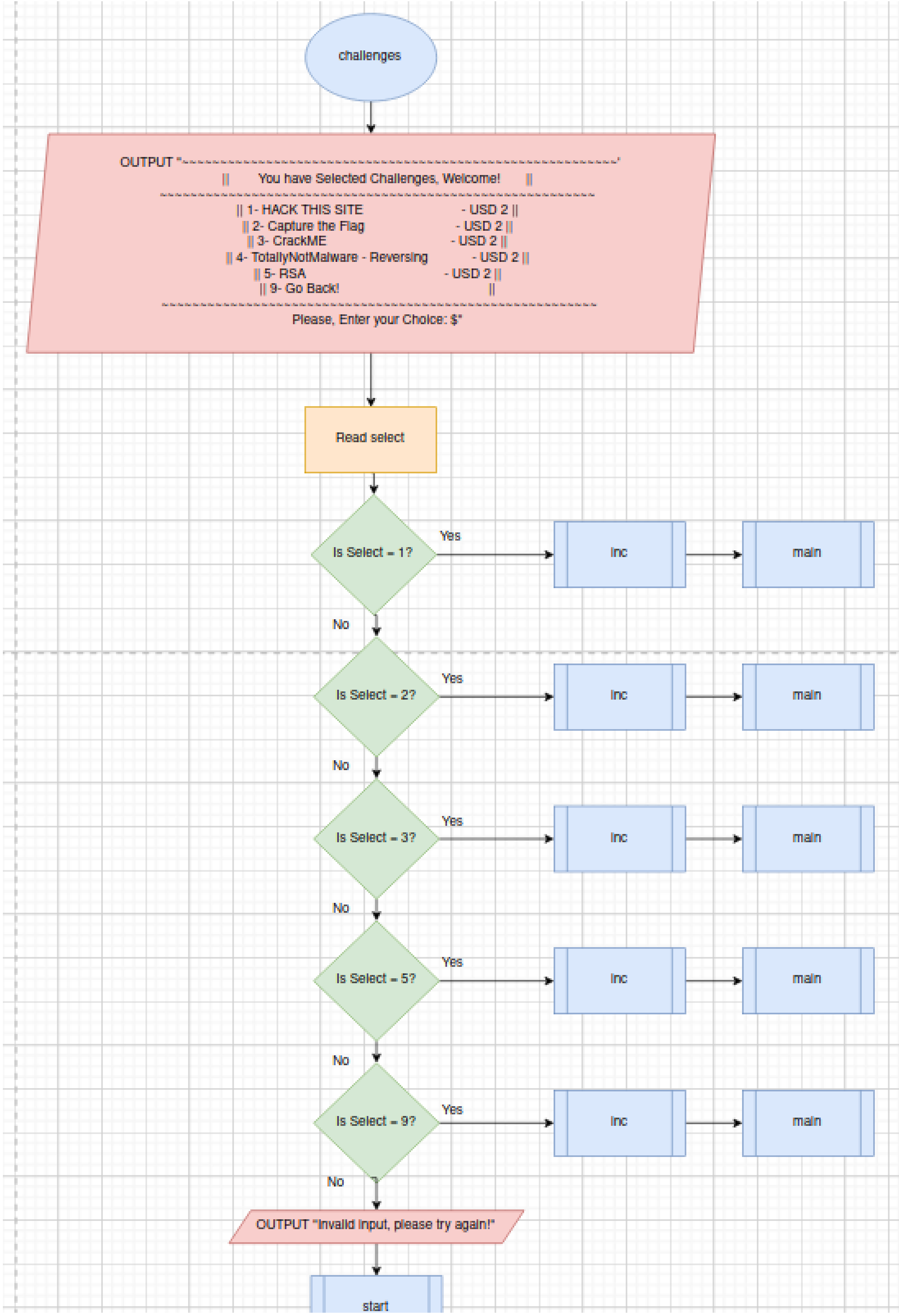

Is Select = 3?
Yes
inc
decSeat
main
No
Is Select = 4?
Yes
inc
decSeat
main
No
Is Select = 5?
Yes
inc
decSeat
main
No
Is Select = 6?
Yes
inc
decSeat
main
No
OUTPUT "Invalid input, please try again!"
main

4.3 Activities Menu

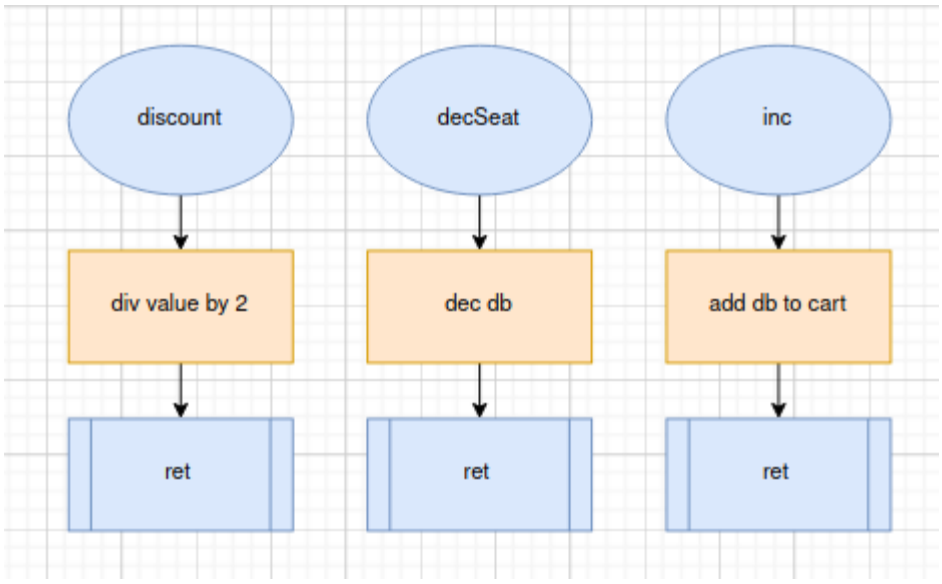




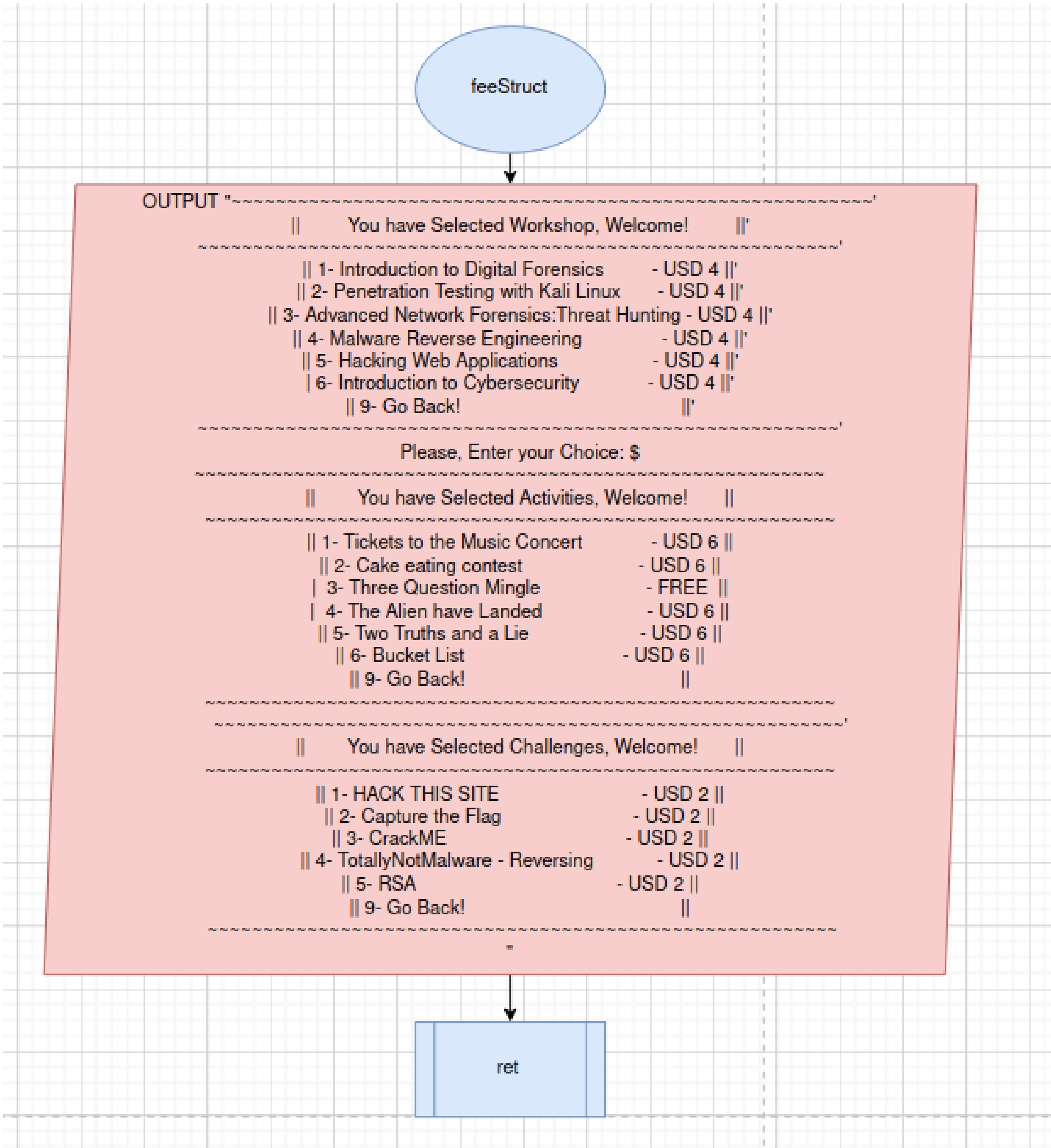
4.4 Challenges


challenges
OUTPUT "~~~~~~
|| You have Selected Challenges, Welcome! ||
~~~~~~
	1- HACK THIS SITE - USD 2	
	2- Capture the Flag - USD 2	
	3- CrackME - USD 2	
	4- TotallyNotMalware - Reversing - USD 2	
	5- RSA - USD 2	
	9- Go Back!	
~~~~~~
Please, Enter your Choice: $"
Read select
Is Select = 1?
Yes
No
Inc
main
Is Select = 2?
Yes
No
Inc
main
Is Select = 3?
Yes
No
Inc
main
Is Select = 5?
Yes
No
Inc
main
Is Select = 9?
Yes
No
Inc
main
OUTPUT "Invalid input, please try again!"
start

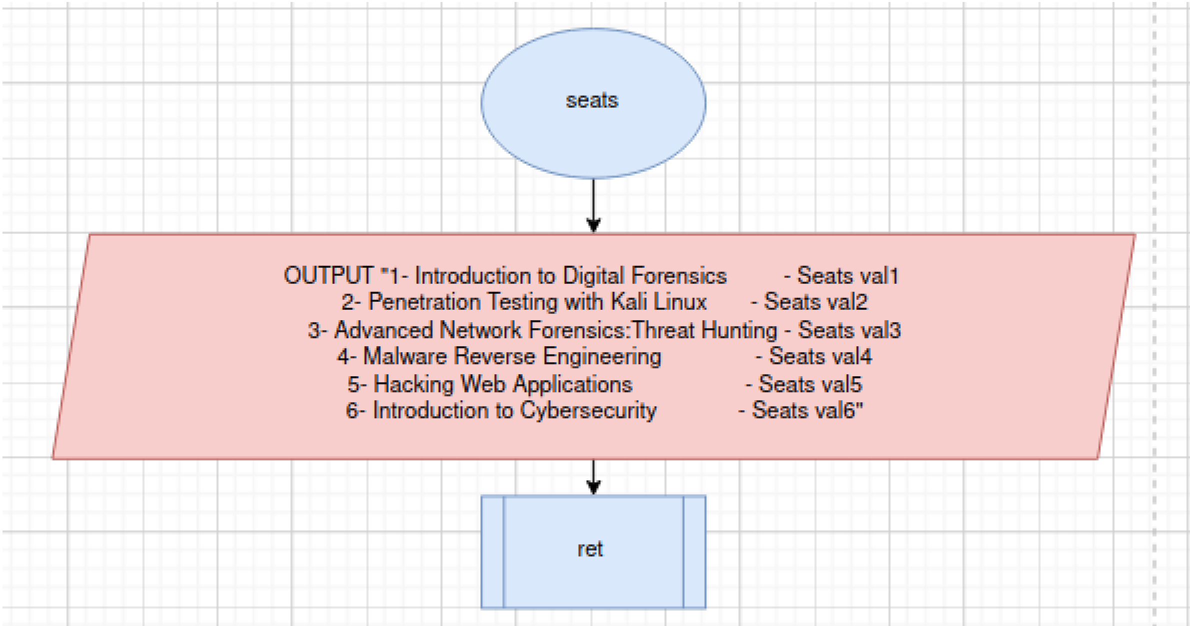
4.5 discount, decSeat, inc



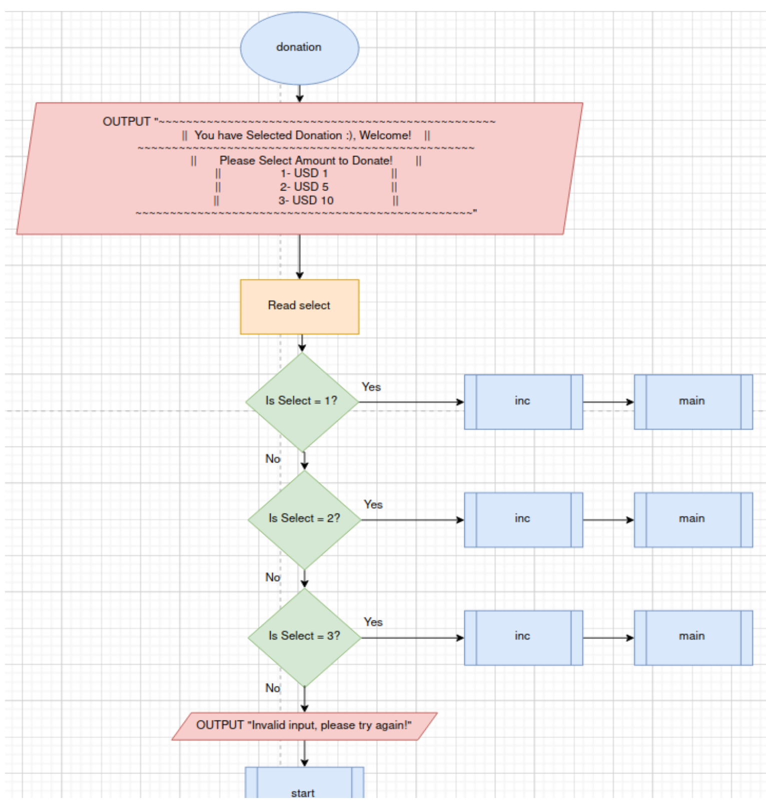
4.6 Fee Structure



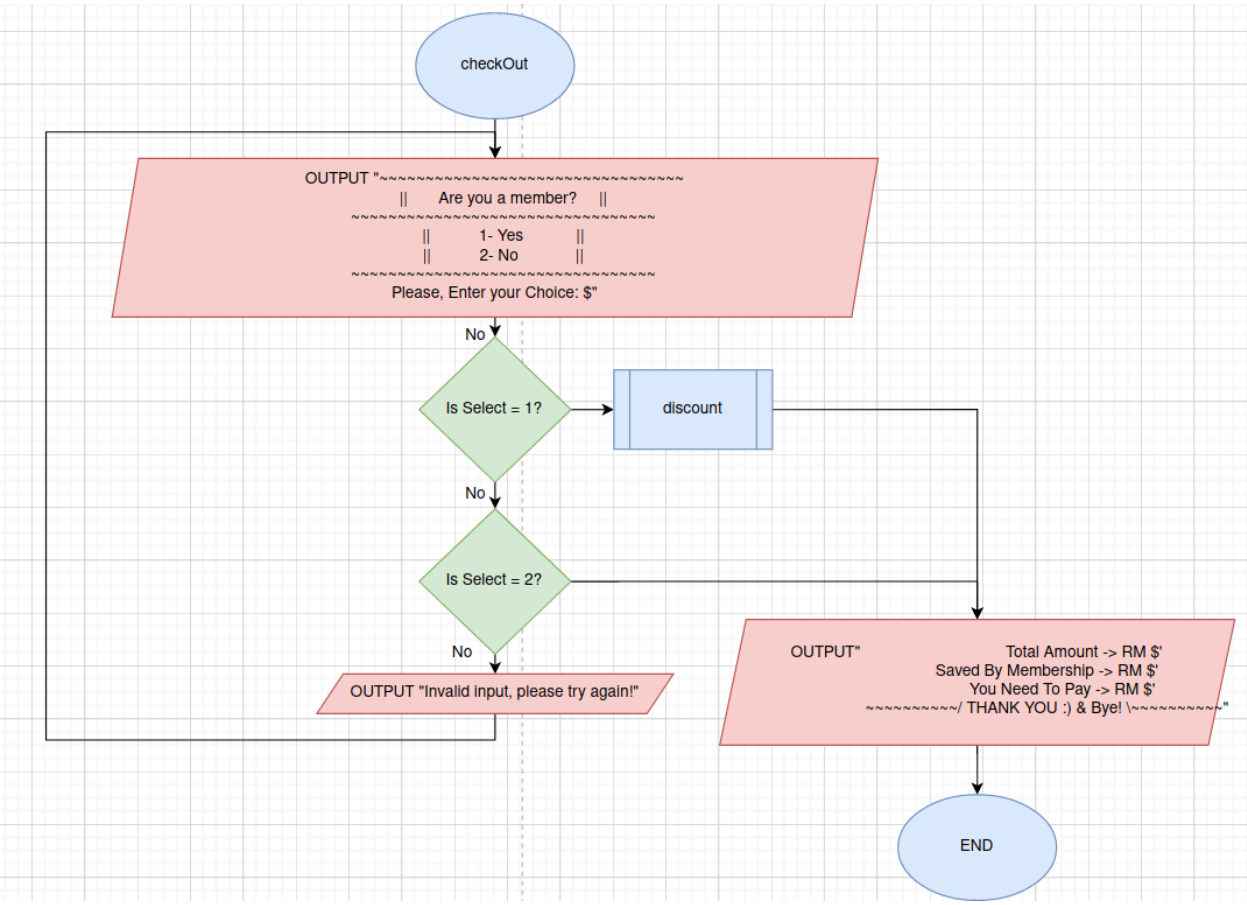
4.7 Show Seats



4.8 Donation

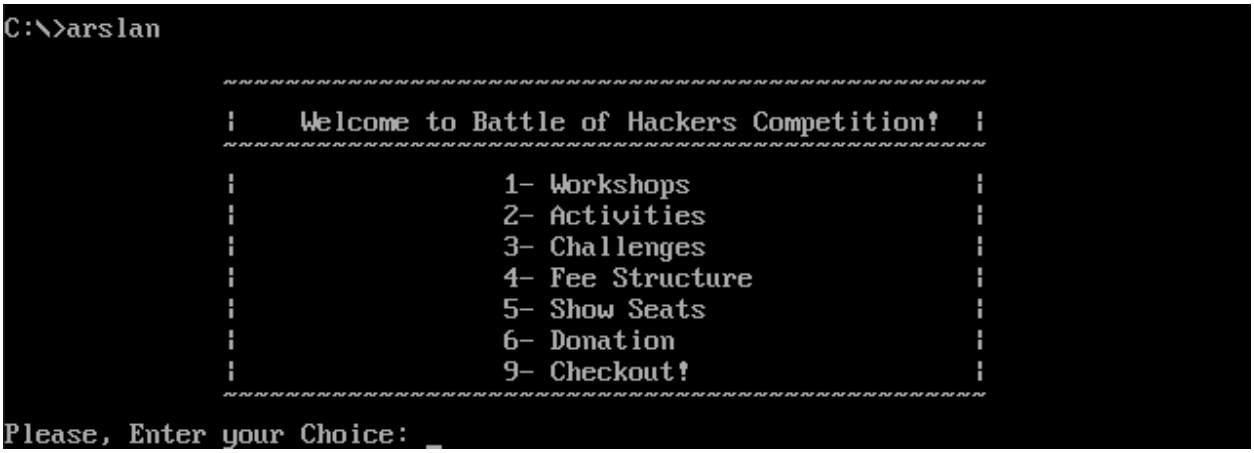


4.9 Check Out

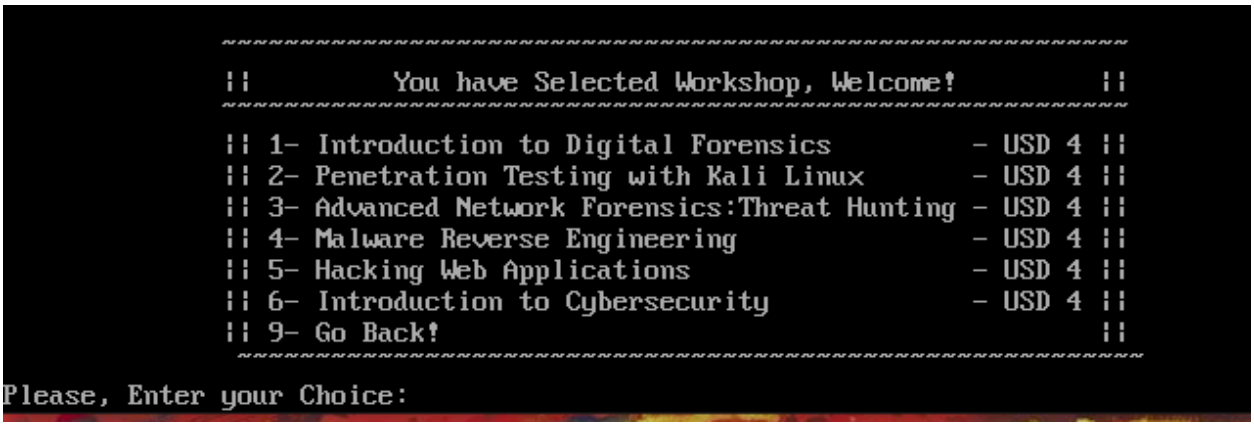

checkOut
OUTPUT "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
|| Are you a member? ||
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
|| 1- Yes ||
|| 2- No ||
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Please, Enter your Choice: $"
No
Is Select = 1?
discount
No
Is Select = 2?
No
OUTPUT "Invalid input, please try again!"
OUTPUT" Total Amount -> RM $'
Saved By Membership -> RM $'
You Need To Pay -> RM $'
~~~~~~~~~~/ THANK YOU :) & Bye! \~~~~~~~~~~"
END
~~~~~~~~~~

5.0 System Screenshot

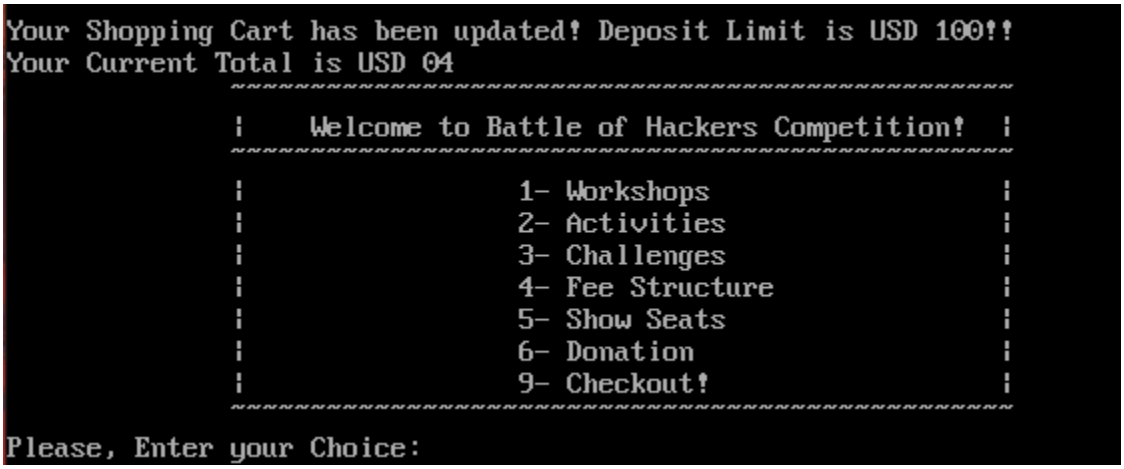
5.1 Welcome Menu



5.2 Workshops Menu

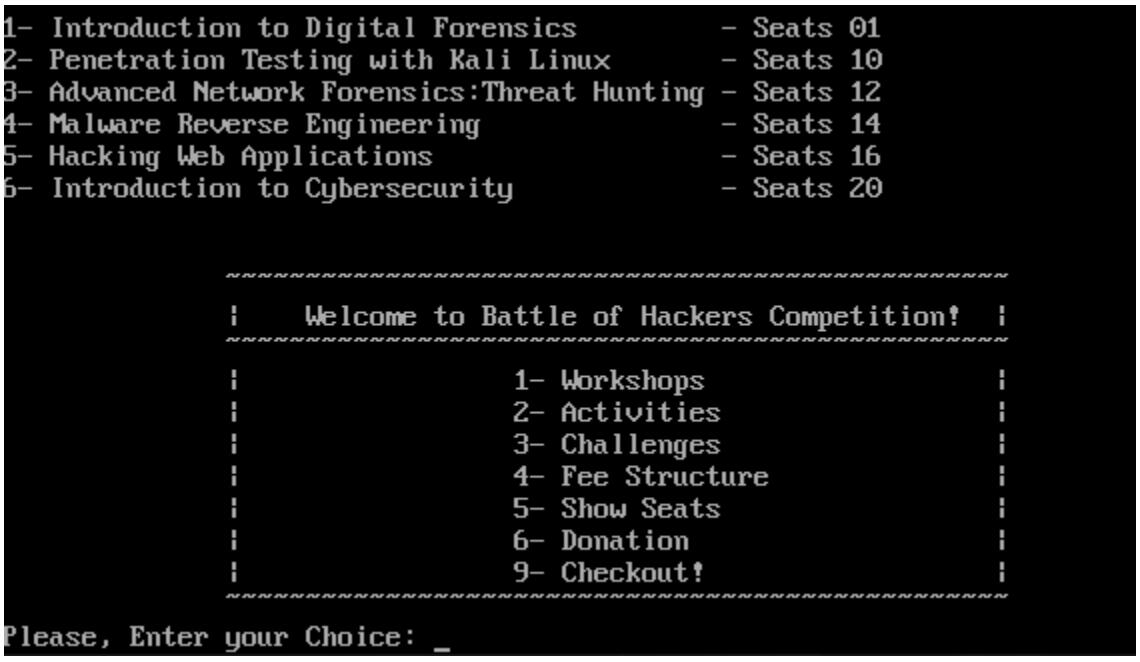


5.3 Adding to Shopping Cart



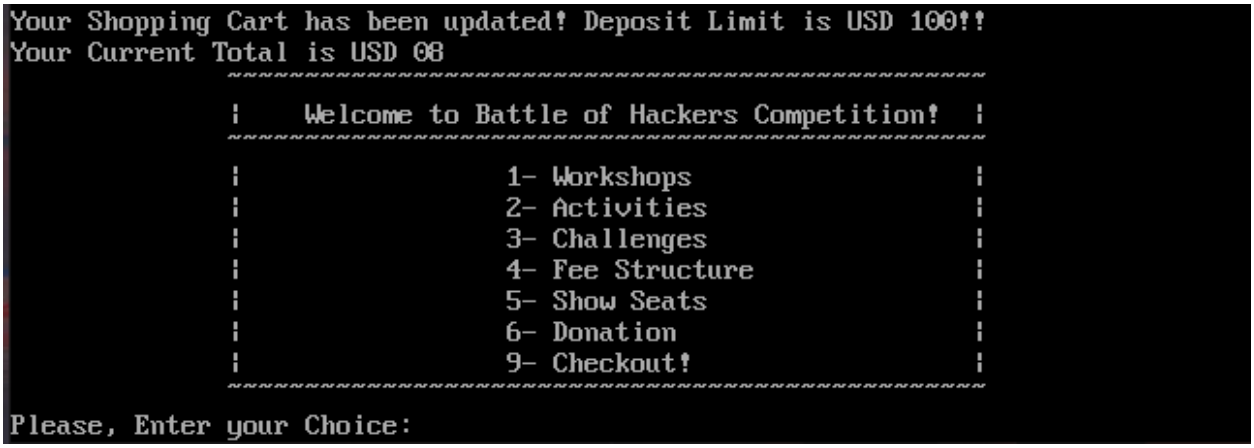
I have selected the first choice which is introduction to digital forensics, as you can see in image above the cart got increment by 4 dollars.

5.4 Remaining Seats



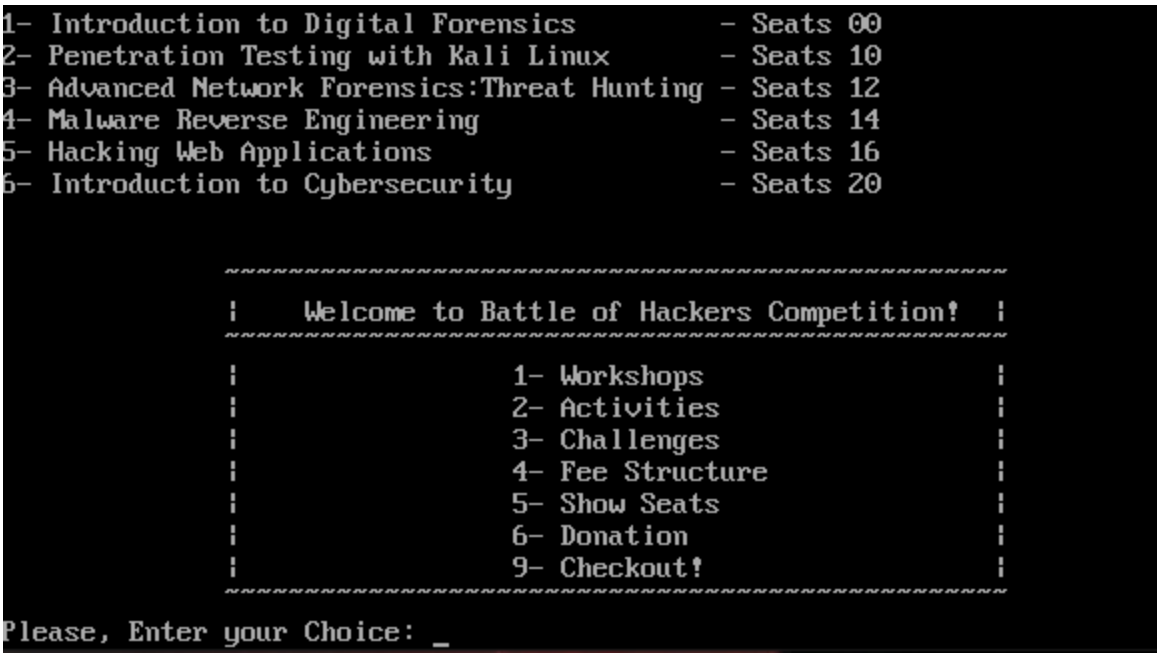
This is the same session. Please take note of seat at Digital Forensics as I am about to select it again.

5.5 Adding to Cart again with Digital Forensics Workshop



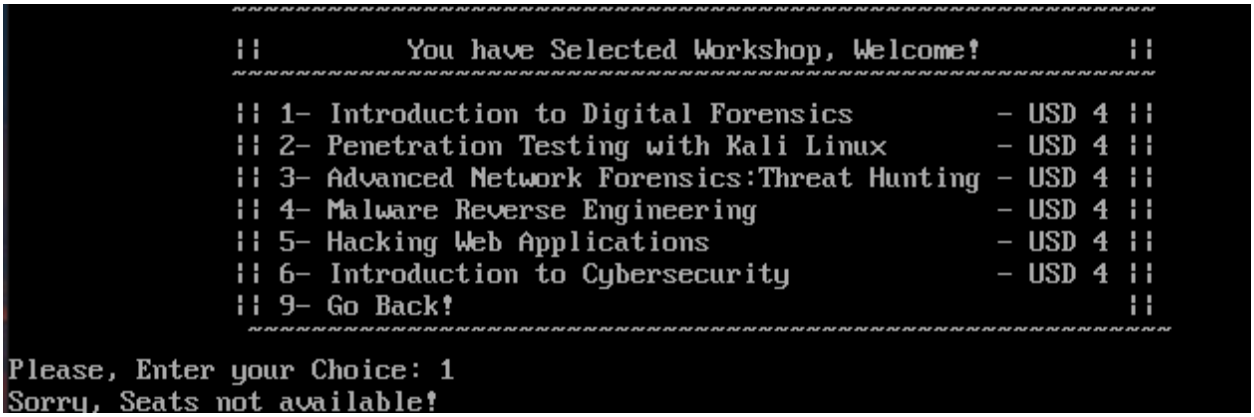
As image above shows that in the same session the cart got increment again by 4 dollars as I have selected the first workshop.

5.6 Remaining seats in the same session



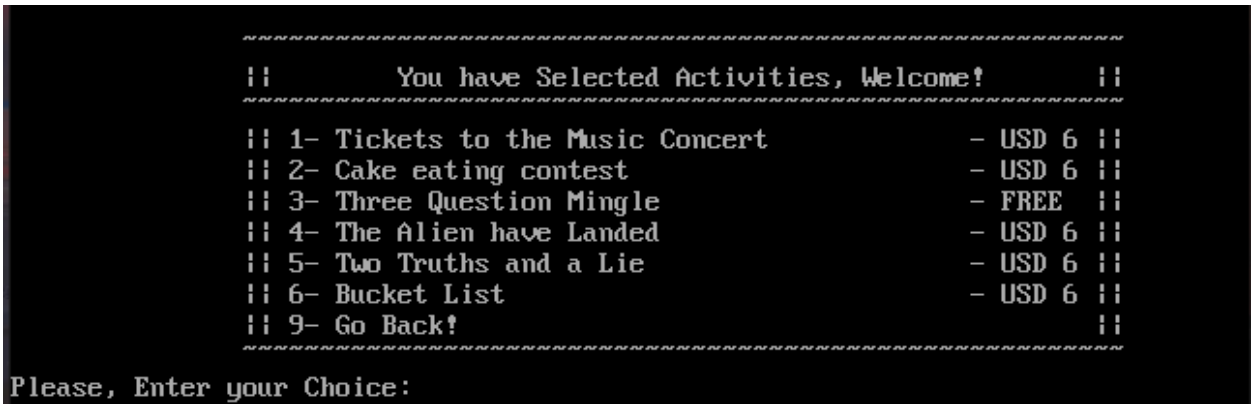
As you can see by looking the above image the seats of Introduction of Digital forensics have been decremented.

5.7 Seats Unavailable



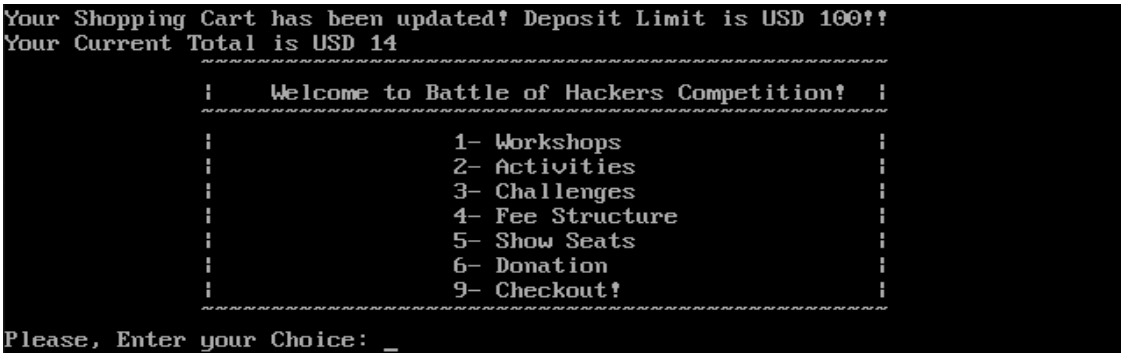
The previous image showed that remaining seats in introduction were 0, so when customer selects choice 1 again. The message “Sorry, Seats not available!” gets printed the console.

5.8 Activities Menu



Please take note that current I have 8 dollars in my cart, and now I will be selecting “The Alien have Landed” activity. The updated cart value should be 14 dollars.

5.9 Updating Cart

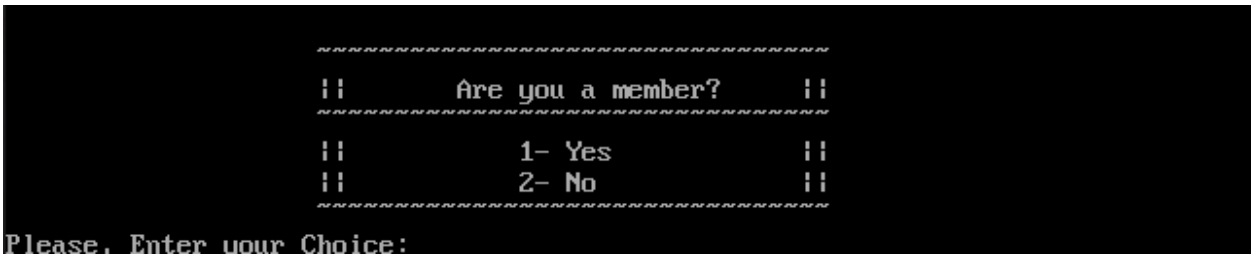


5.10 Challenges Menu

Cart total currently is 14 Dollars, and I am going to select 'TotallyNotMalware'. Total should become 16 dollars.

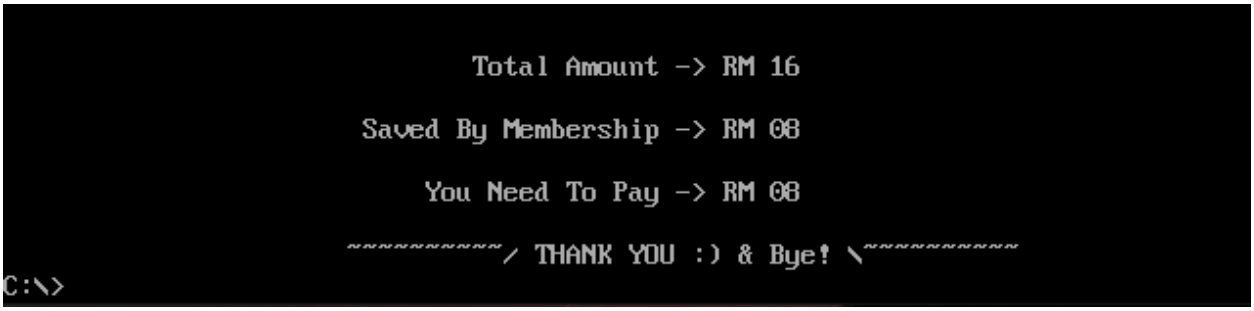
5.11 Incremented Cart

5.12 Check Out

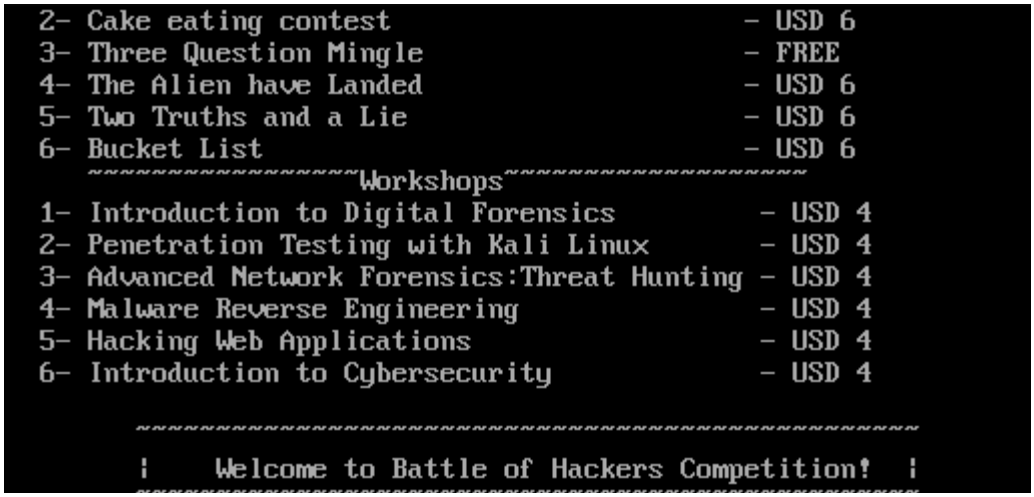


After the user has selected check out the system will ask whether you are a member or not. If the user is a member then 50 Percent discount will given to the member. I will be pressing '1' and selecting 'Yes'. My Shopping Cart has the Value of 16 dollars as it can seen in previous image. Therefore, a person with membership only needs to pay 8 dollars.

5.13 Receipt



5.14 Additional Feature: Fee Structure



This feature is not a part requirements, and it just displays the list of all workshops, activities, challenges and their prices. In other words, its just like a menu in a restaurant.

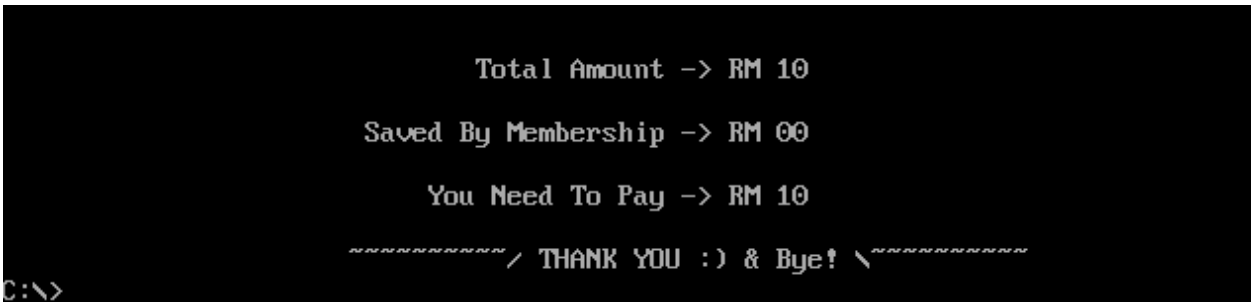
5.15 Additional Feature: Donation

This is the Donation Screen, the customer is able donate his/her money to the cause. This is a new session, so cart will be zero right now. I will be selecting 10 dollars.

5.16 Additional Feature: Donation added to Cart

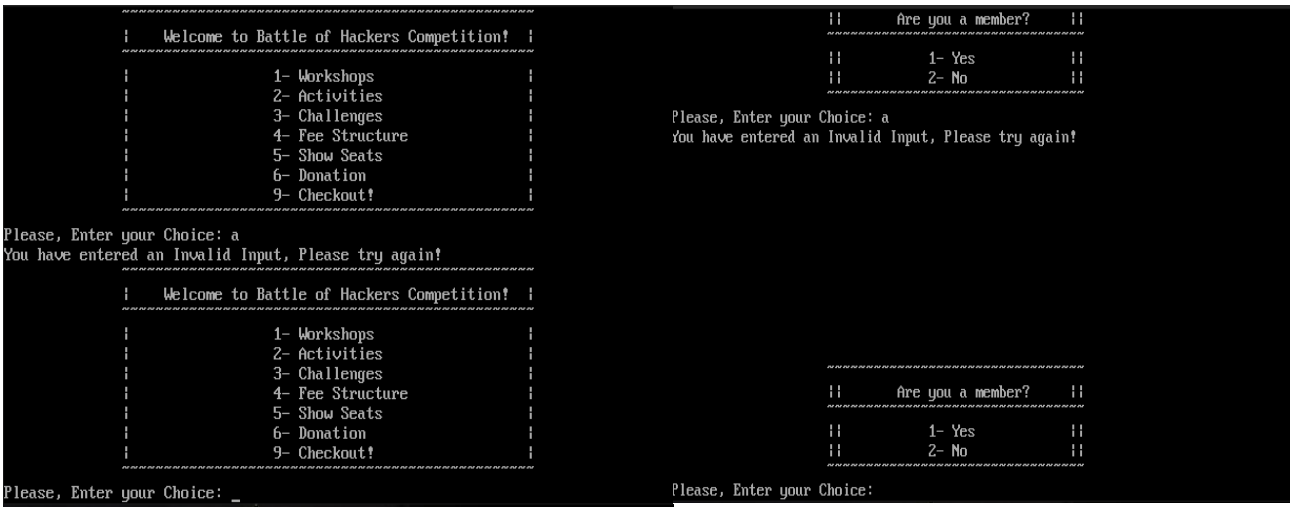
As we can see the cart has gotten incremented, and please note that this function works along side other calculations of the system. This was illustrated in my presentation. I will be selecting check out next. At the membership menu I will be selecting 'No' meaning the customer does not have membership.

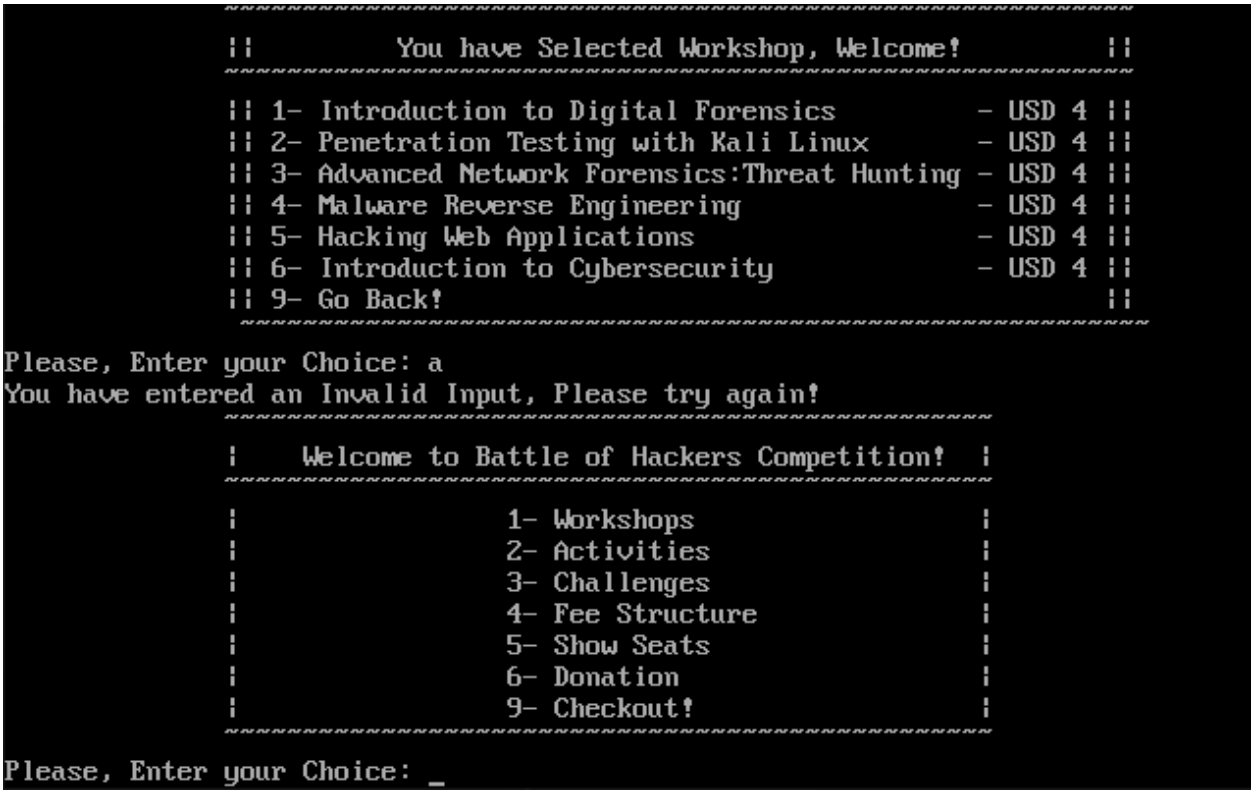
5.17 Check Out



As seen above, if customer does not membership the full amount needs to be payed.

5.18 System Validation





I have implemented validations such as shown above through-out my program.

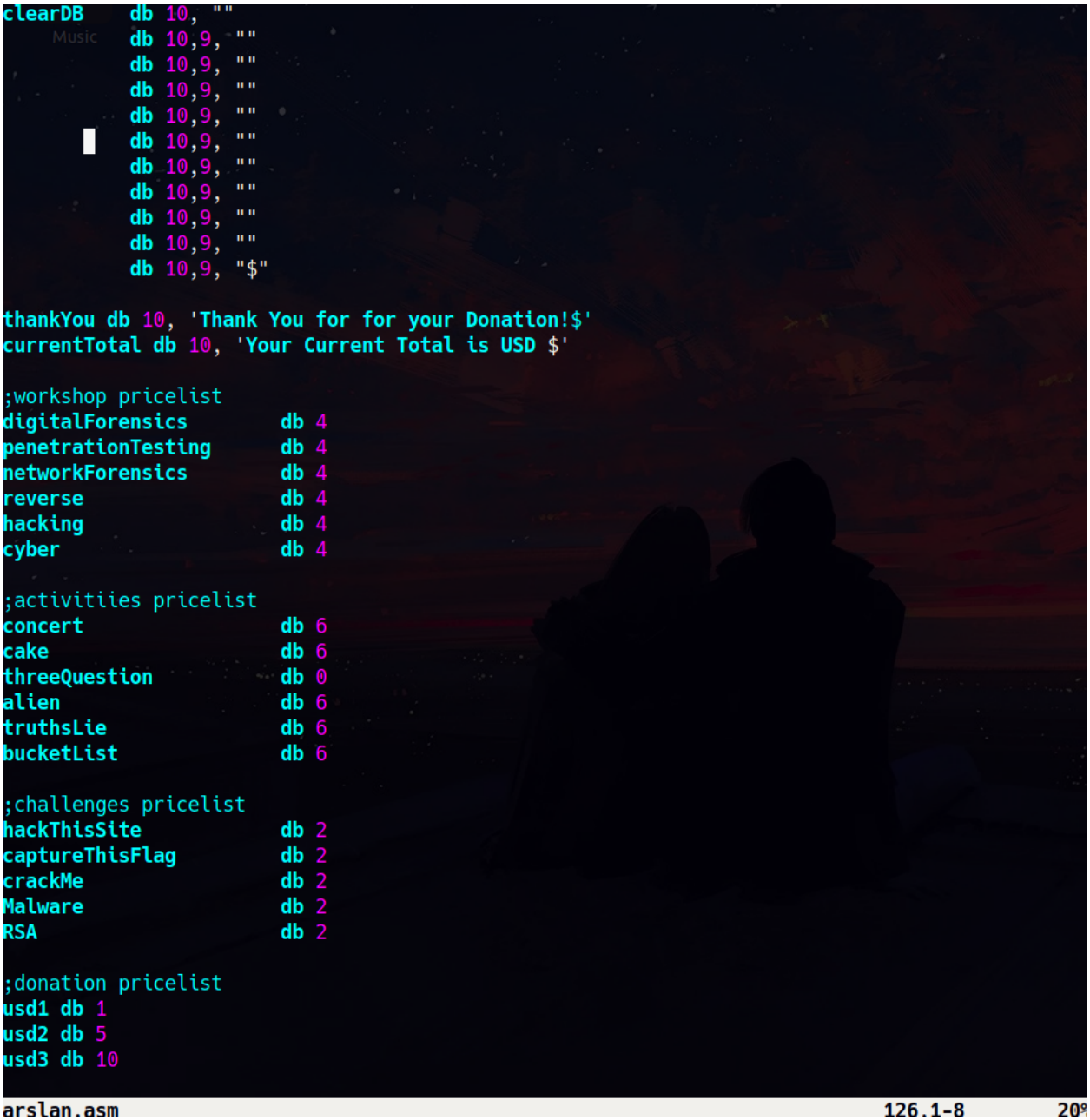
6.0 Source Code

6.1 Database

```
;author: Arslan Khurram
;date: 01/02/2022
;TP number: TP058344
;Program Name: APU Cybersecurity Club's Cash Register Program

.model small
.stack 100h
.data

welcome db 10, '                        ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~'
        db 10, '                        |    Welcome to Battle of Hackers Competition!   |'
        db 10, '                        ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~'
        db 10, '                        |                 1- Workshops                   |'
        db 10, '                        |                 2- Activities                  |'
        db 10, '                        |                 3- Challenges                  |'
        db 10, '                        |                 4- Fee Structure               |'
        db 10, '                        |                 5- Show Seats                  |'
        db 10, '                        |                 6- Donation                    |'
        db 10, '                        |                 9- Checkout!                   |'
        db 10, '                        ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~'
        db 10,13, 'Please, Enter your Choice: $'

error db 10, 'You have entered an Invalid Input, Please try again! $'
addCart db 10, 'Your Shopping Cart has been updated! Deposit Limit is USD 100!! $'
workshopDB db 10, '                    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~'
           db 10, '                    ||         You have Selected Workshop, Welcome!           ||'
           db 10, '                    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~'
           db 10, '                    || 1- Introduction to Digital Forensics         - USD 4 ||'
           db 10, '                    || 2- Penetration Testing with Kali Linux       - USD 4 ||'
           db 10, '                    || 3- Advanced Network Forensics:Threat Hunting - USD 4 ||'
           db 10, '                    || 4- Malware Reverse Engineering               - USD 4 ||'
           db 10, '                    || 5- Hacking Web Applications                  - USD 4 ||'
           db 10, '                    || 6- Introduction to Cybersecurity             - USD 4 ||'
           db 10, '                    || 9- Go Back!                                          ||'
           db 10, '                     ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~'
           db 10,13, 'Please, Enter your Choice: $'
```

```
activitiesDB db 10, '                ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
             db 10, '                ||        You have Selected Activities, Welcome!        ||
             db 10, '                ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
             db 10, '                || 1- Tickets to the Music Concert                - USD 6 ||
             db 10, '                || 2- Cake eating contest                         - USD 6 ||
             db 10, '                || 3- Three Question Mingle                       - FREE  ||
             db 10, '                || 4- The Alien have Landed                       - USD 6 ||
             db 10, '                || 5- Two Truths and a Lie                        - USD 6 ||
             db 10, '                || 6- Bucket List                                 - USD 6 ||
             db 10, '                || 9- Go Back!                                            ||
             db 10, '                ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
             db 10, 'Please, Enter your Choice: $'

challengesDB db 10, '              ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~'
             db 10, '              ||        You have Selected Challenges, Welcome!        ||'
             db 10, '              ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~'
             db 10, '              || 1- HACK THIS SITE                              - USD 2 ||'
             db 10, '              || 2- Capture the Flag                            - USD 2 ||'
             db 10, '              || 3- CrackME                                     - USD 2 ||'
             db 10, '              || 4- TotallyNotMalware - Reversing               - USD 2 ||'
             db 10, '              || 5- RSA                                         - USD 2 ||'
             db 10, '              || 9- Go Back!                                            ||'
             db 10, '              ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~'
             db 10, 'Please, Enter your Choice: $'

feeDB      db 10,9, 'You have Selected Fee Structure, Welcome!'
           db 10,9, '   ~~~~~~~~~~~~~~~~~~Challenges~~~~~~~~~~~~~~~~~~~'
           db 10,9, '1- HACK THIS SITE                          - USD 2'
           db 10,9, '2- Capture the Flag                        - USD 2'
           db 10,9, '3- CrackME                                 - USD 2'
           db 10,9, '4- TotallyNotMalware - Reversing           - USD 2'
           db 10,9, '5- RSA                                     - USD 2'
           db 10,9, '   ~~~~~~~~~~~~~~~~~~Activities~~~~~~~~~~~~~~~~~~~'
           db 10,9, '1- Tickets to the Music Concert            - USD 6'
           db 10,9, '2- Cake eating contest                     - USD 6'
           db 10,9, '3- Three Question Mingle                   - FREE'
           db 10,9, '4- The Alien have Landed                   - USD 6'
           db 10,9, '5- Two Truths and a Lie                    - USD 6'
           db 10,9, '6- Bucket List                             - USD 6'
           db 10,9, '   ~~~~~~~~~~~~~~~~~~Workshops~~~~~~~~~~~~~~~~~~~'
           db 10,9, '1- Introduction to Digital Forensics        - USD 4'
```

```
          db 10,9, '1- Introduction to Digital Forensics         - USD 4'
          db 10,9, '2- Penetration Testing with Kali Linux       - USD 4'
          db 10,9, '3- Advanced Network Forensics:Threat Hunting - USD 4'
          db 10,9, '4- Malware Reverse Engineering               - USD 4'
          db 10,9, '5- Hacking Web Applications                  - USD 4'
          db 10,9, '6- Introduction to Cybersecurity             - USD 4$'

seats1DB db 10, '1- Introduction to Digital Forensics         - Seats $'
seats2DB db 10, '2- Penetration Testing with Kali Linux       - Seats $'
seats3DB db 10, '3- Advanced Network Forensics:Threat Hunting - Seats $'
seats4DB db 10, '4- Malware Reverse Engineering               - Seats $'
seats5DB db 10, '5- Hacking Web Applications                  - Seats $'
seats6DB db 10, '6- Introduction to Cybersecurity             - Seats $'

seatsZero db 10, 'Sorry, Seats not available!$'

;workshop seats
dfS     db 2
ptS     db 10
nfS     db 12
rS      db 14
hS      db 16
cyS     db 20


membershipDB db 10, '                  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~'
             db 10, '                  ||        Are you a member?     ||'
             db 10, '                  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~'
             db 10, '                  ||            1- Yes            ||'
             db 10, '                  ||            2- No             ||'
             db 10, '                  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~'
             db 10, 'Please, Enter your Choice: $'
donationDB db 10, '               ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~'
           db 10, '               ||  You have Selected Donation :), Welcome!    ||'
           db 10, '               ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~'
           db 10, '               ||       Please Select Amount to Donate!       ||'
           db 10, '               ||                1- USD 1                     ||'
           db 10, '               ||                2- USD 5                     ||'
           db 10, '               ||                3- USD 10                    ||'
           db 10, '               ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~'
```



```
clearDB     db 10, ""
Music       db 10,9, ""
            db 10,9, ""
            db 10,9, ""
            db 10,9, ""
            db 10,9, ""
            db 10,9, ""
            db 10,9, ""
            db 10,9, ""
            db 10,9, ""
            db 10,9, "$"

thankYou db 10, 'Thank You for for your Donation!$'
currentTotal db 10, 'Your Current Total is USD $'

;workshop pricelist
digitalForensics        db 4
penetrationTesting      db 4
networkForensics        db 4
reverse                 db 4
hacking                 db 4
cyber                   db 4

;activitiies pricelist
concert                 db 6
cake                    db 6
threeQuestion           db 0
alien                   db 6
truthsLie               db 6
bucketList              db 6

;challenges pricelist
hackThisSite            db 2
captureThisFlag         db 2
crackMe                 db 2
Malware                 db 2
RSA                     db 2

;donation pricelist
usd1 db 1
usd2 db 5
usd3 db 10
```

arslan.asm 126,1-8 20%

```
;global
total db 0
minus db 0
pay   db 0
totalCheckOut   db 10, '                              Total Amount -> RM $'
saveMember      db 10, '                           Saved By Membership -> RM $'
needToPay       db 10, '                               You Need To Pay -> RM $'
exitMsg         db 10,'                          ~~~~~~~~~~/ THANK YOU :) & Bye! \~~~~~~~~~~$'
empty           db 10, "$"

.code
```

6.2 Macros

```
.code

ShowMessage Macro Mess  ;Macro for Display

        mov ah, 09h
        mov dx, offset Mess
        int 21h
EndM
GetMessage Macro get    ;get input from user
        mov ah,1 ;read char
        int 21h
EndM
ShowTotal Macro db ;Display db values, 2 digits
        mov al,db
        mov bl,0
        add al,bl

        aam
        add ax,3030h

        mov dh,al
        mov dl,ah

        mov ah,2
        int 21h

        mov dl,dh
        mov ah,2
        int 21h
EndM
DecSeats Macro val      ;get input from user
        mov al,val ;read char
        dec al
        mov val, al
EndM
```

I tried to reduce and optimize my code with help of macro as much I could. Please check the comments for further details.

6.3 Welcome Menu & Main Proc

```
Main proc
    mov ax,@data
    mov ds,ax

    welcomePage:
            ShowMessage welcome ;calls Macro to display with parameter welcome
            GetMessage

            cmp al, 49 ;comparing input to 49 Decimal which equals to 1 char
            jne two1 ;if comparison is false then jump to activities
            call workshop  ;else it proceeds to workshop function
            jmp loop1   ;after performing workshop start at the welcome screen
    two1:
            cmp al, 50 ;equal to 2
            jne three1
            call activities
            jmp loop1
    three1:
            cmp al, 51 ;equal to 3
            jne four1
            call challenges
            jmp loop1
    four1:
            cmp al, 52 ;equal to 4
            jne five1
            ShowMessage clearDB
            ShowMessage feeDB
            ShowMessage empty
            jmp loop1
    five1:
            cmp al, 53 ;equal to 5
            jne six1
            call showSeats
            jmp loop1
    six1:
            cmp al, 54 ;equal to 6
            jne exit1
            call donation
            jmp loop1
    exit1:
            cmp al, 57 ;equal to 9
            jne invalid1
            jmp end2
```

```
        invalid1:
                mov ah, 09h
                mov dx, offset error
                int 21h
                jmp loop1
        loop1:
                loop welcomePage ;goes back to welcome
        end2:
                call member
                call displayTotal
                mov ah,4ch ;teriminating dos
                int 21h
Main endp
```

Main Proc prints the menu on the screen and takes input from the user. After a successful call it keeps on repeating until user selects checkout.

6.4 Workshop Proc

```
workshop proc
        mov ax,@data
        mov ds,ax

        workshopPage:
                ShowMessage clearDB
                ShowMessage workshopDB ;calls macro to display message
                GetMessage

                cmp al, 49 ;comparing input to 49 Decimal which equals to 1 char
                jne two2 ;if comparison is false then jump to two
                Call SeatCheckdfS ;checking whether seats are zero
                mov al, digitalForensics
                call increment
                DecSeats dfS ;decrement seat by 1
                ret     ; to repeat display this screen
        two2:
                cmp al, 50 ;equal to 2
                jne three2
                Call SeatCheckptS
                mov al, penetrationTesting ;getting value from db and mov to al
                call increment
                DecSeats ptS ;decrement seat by 1
                ret
        three2:
                cmp al, 51 ;equal to 3
                jne four2
                Call SeatChecknfS
                mov al, networkForensics
                call increment
                DecSeats nfS ;decrement seat by 1
                ret
        four2:
                cmp al, 52
                jne five2
                Call SeatCheckrS
                mov al, reverse
                call increment
                DecSeats rS ;decrement seat by 1
                ret
        five2:
```

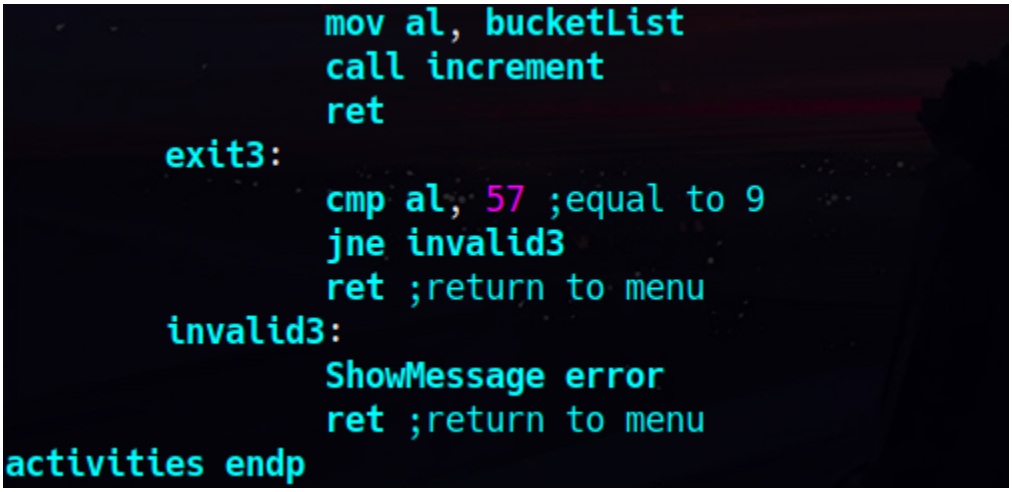
```
        cmp al, 53 ;equal to 5
        jne six2
        Call SeatCheckhS
        mov al, hacking
        call increment
        DecSeats hS ;decrement seat by 1
        ret
six2:
        cmp al, 54 ;equal to 6
        jne exit2
        Call SeatCheckcyS
        mov al, cyber
        call increment
        DecSeats cyS ;decrement seat by 1
        ret
exit2:
        cmp al, 57 ;equal to 9
        jne invalid2
        ShowMessage clearDB
        ret ;return to menu
invalid2:
        ShowMessage error
        ret ;return to menu
workshop endp
```

6.5 Increment Proc

```
increment proc
        add al, total ;db value gets added to the total
        mov total, al ;updating total
        mov pay, al ;inc total
        ShowMessage clearDB
        ShowMessage addCart
        ShowMessage currentTotal
        ShowTotal total
        ret ;return to the call
increment endp
```

6.6 Activities Proc

```
activities proc
        mov ax,@data
        mov ds,ax

        activitiesPage:
                ShowMessage clearDB
                ShowMessage activitiesDB ;calls macro to display message
                GetMessage

                cmp al, 49 ;comparing input to 49 Decimal which equals to 1 char
                jne two3 ;if comparison is false then jump to two
                mov al, concert
                call increment
                ret    ; to repeat display this screen
        two3:
                cmp al, 50 ;equal to 2
                jne three3
                mov al, cake ;getting value from db and mov to al
                call increment
                ret
        three3:
                cmp al, 51 ;equal to 3
                jne four3
                mov al, threeQuestion
                call increment
                ret
        four3:
                cmp al, 52
                jne five3
                mov al, alien
                call increment
                ret
        five3:
                cmp al, 53 ;equal to 5
                jne six3
                mov al, truthsLie
                call increment
                ret
        six3:
                cmp al, 54 ;equal to 6
                jne exit3
                mov al, bucketList
```



```
                mov al, bucketList
                call increment
                ret
        exit3:
                cmp al, 57 ;equal to 9
                jne invalid3
                ret ;return to menu
        invalid3:
                ShowMessage error
                ret ;return to menu
activities endp
```

6.9 Challenges Proc

```
challenges proc
        mov ax,@data
        mov ds,ax

        challengesPage:
                ShowMessage clearDB
                ShowMessage challengesDB ;calls macro to display message
                GetMessage

                cmp al, 49 ;comparing input to 49 Decimal which equals to 1 char
                jne two4 ;if comparison is false then jump to two
                mov al, hackThisSite
                call increment
                ret    ; to repeat display this screen
        two4:
                cmp al, 50 ;equal to 2
                jne three4
                mov al, captureThisFlag ;getting value from db and mov to al
                call increment
                ret
        three4:
                cmp al, 51 ;equal to 3
                jne four4
                mov al, crackMe
                call increment
                ret
        four4:
                cmp al, 52
                jne five4
                mov al, malware
                call increment
                ret
        five4:
                cmp al, 53 ;equal to 5
                jne exit4
                mov al, RSA
                call increment
                ret
        exit4:
                cmp al, 57 ;equal to 9
                jne invalid4
                ret ;return to menu
        invalid4:
                ShowMessage error
                ret ;return to menu
challenges endp
```

6.10 DisplayTotal Proc – Receipt Message

```
displayTotal proc
        mov ax,@data
        mov ds,ax

        showMessage totalCheckOut ;display total message
        ShowTotal total ;macro shows total
        ShowMessage empty ;newline

        showMessage saveMember
        ShowTotal minus
        ShowMessage empty

        showMessage needToPay
        ShowTotal pay
        ShowMessage empty

        showMessage exitMsg ;prints thank you
        ret ;return to the call
displayTotal endp
```

6.11 Member Proc – Membership Menu

```
member proc
        mov ax,@data
        mov ds,ax

        memberPage:
                ShowMessage clearDB
                ShowMessage membershipDB ;calls macro to display message
                GetMessage

                cmp al, 49 ;comparing input to 49 Decimal which equals to 1 char
                jne two5 ;if comparison is false then jump to two
                ;if he is a member - give discount 50%
                ShowMessage clearDB
                call discount ;do discount
                ret    ; to repeat display this screen
        two5:
                cmp al, 50 ;equal to 2
                jne invalid5
                ShowMessage clearDB
                ret ; if not a member it just return back to the call without changes
        invalid5:
                ShowMessage error
                jmp loop5
        loop5:  loop memberPage
member endp
```

6.12 Discount Proc

```
discount proc
        mov al, total
        mov bl ,1
        mul bl ;making ax register

        mov bl, 2 ;divide by 2 for discount
        div bl ;with ax register from mul
        mov minus,al ;return val

        mov pay,al ;give to pay

        ret
discount endp
```

6.13 Show Seats Proc

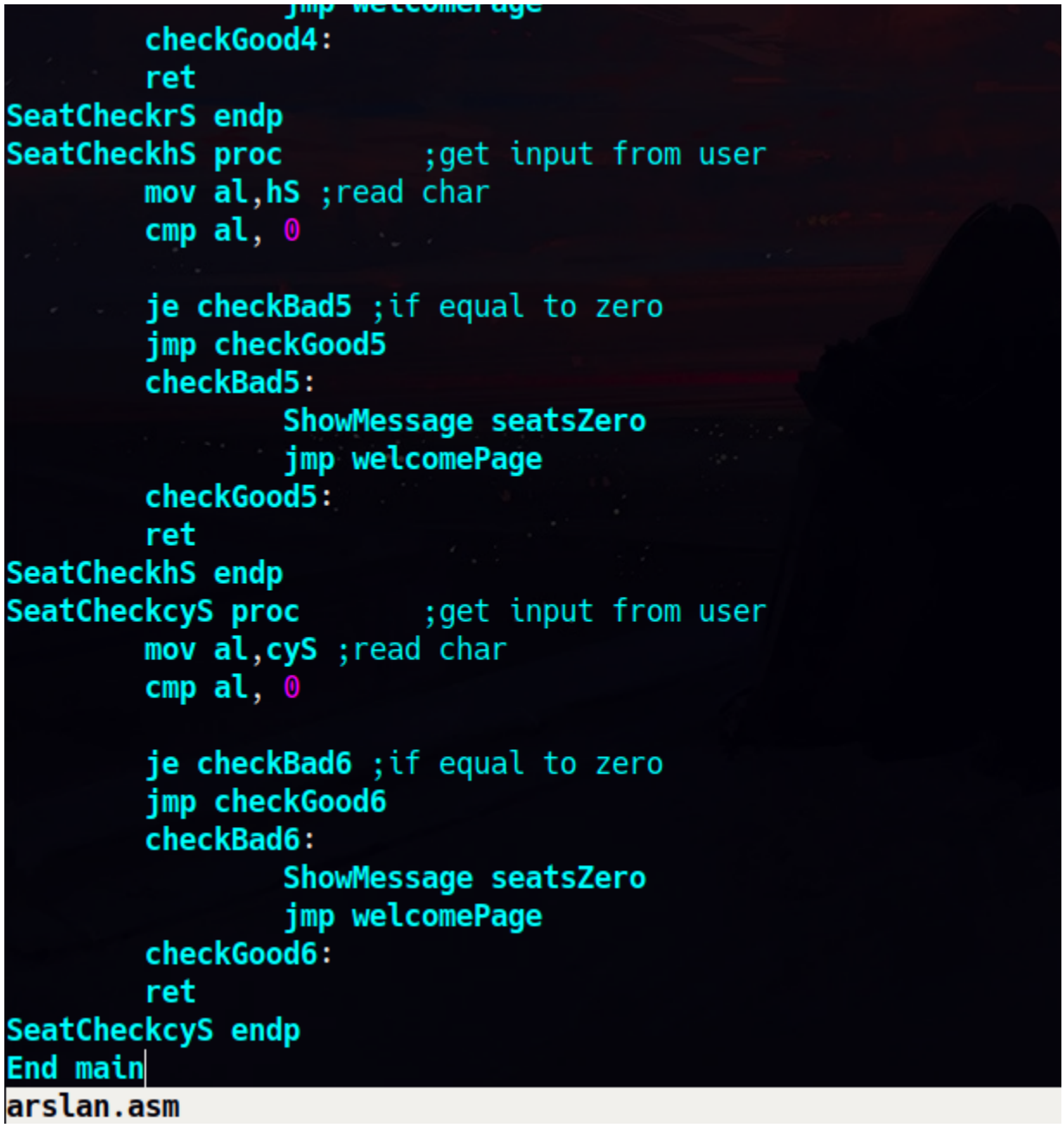
```
showSeats proc
        ShowMessage empty
        ShowMessage seats1DB
        ShowTotal dfS
        ShowMessage seats2DB
        ShowTotal ptS
        ShowMessage seats3DB
        ShowTotal nfS
        ShowMessage seats4DB
        ShowTotal rS
        ShowMessage seats5DB
        ShowTotal hS
        ShowMessage seats6DB
        ShowTotal cyS
        ShowMessage empty
        ShowMessage empty
        ret
showSeats endp
```

6.14 Donation Proc – Display donation Page

```
donation proc
        mov ax,@data
        mov ds,ax

        donationPage:
                ShowMessage clearDB
                ShowMessage donationDB ;calls macro to display message
                GetMessage

                cmp al, 49 ;comparing input to 49 Decimal which equals to 1 char
                jne two6 ;if comparison is false then jump to two
                mov al, usd1
                call increment
                ShowMessage thankYou
                ret    ; to repeat display this screen
        two6:
                cmp al, 50 ;equal to 2
                jne three6
                mov al, usd2 ;getting value from db and mov to al
                call increment
                ShowMessage thankYou
                ret
        three6:
                cmp al, 51 ;equal to 3
                jne invalid6
                mov al, usd3
                call increment
                ShowMessage thankYou
                ret
        invalid6:
                ShowMessage error
                ret ;return to menu
donation endp
```

6.15 Seat Check Proc

```
SeatCheckdfS proc          ;get input from user
        mov al,dfS ;read char
        cmp al, 0

        je checkBad1 ;if equal to zero
        jmp checkGood1
        checkBad1:
                ShowMessage seatsZero
                jmp welcomePage
        checkGood1:
        ret
SeatCheckdfS endp
SeatCheckptS proc          ;get input from user
        mov al,ptS ;read char
        cmp al, 0

        je checkBad2 ;if equal to zero
        jmp checkGood2
        checkBad2:
                ShowMessage seatsZero
                jmp welcomePage
        checkGood2:
        ret
SeatCheckptS endp
SeatCheecknfS proc         ;get input from user
        mov al,nfS ;read char
        cmp al, 0

        je checkBad3 ;if equal to zero
        jmp checkGood3
        checkBad3:
                ShowMessage seatsZero
                jmp welcomePage
        checkGood3:
        ret
SeatCheecknfS endp
SeatCheckrS proc           ;get input from user
        mov al,rS ;read char
        cmp al, 0

        je checkBad4 ;if equal to zero
        jmp checkGood4
        checkBad4:
```



```
        checkGood4:
        ret
SeatCheckrS endp
SeatCheckhS proc        ;get input from user
        mov al,hS ;read char
        cmp al, 0

        je checkBad5 ;if equal to zero
        jmp checkGood5
        checkBad5:
                ShowMessage seatsZero
                jmp welcomePage
        checkGood5:
        ret
SeatCheckhS endp
SeatCheckcyS proc       ;get input from user
        mov al,cyS ;read char
        cmp al, 0

        je checkBad6 ;if equal to zero
        jmp checkGood6
        checkBad6:
                ShowMessage seatsZero
                jmp welcomePage
        checkGood6:
        ret
SeatCheckcyS endp
End main
```

arslan.asm

7.0 Limitation

The only limitation in my system is that it cannot display more than 2 digits in receipt.

8.0 Conclusion & Self – Reflection

Computer Systems Low Level Techniques module has taught me that how hard it is to do simple tasks in assembly language. If this assignment was written in a high level language it would have taken only a few hours to complete the system. I have learned how easy has the compilers and interpreters made our life. I hope to learn more about assembly in my free time. Also I have tried with best ability to finish the assignment but I still wish that I had more time so I could implement and display more than 2 characters and decimals spaces on the display. During the research, I have learn how important is assembly to the world and how it has been implemented in world's most important systems. I am thankful to my teacher, I would not be able to do this assignment without her help and guidance.

9.0 References

Streib, J. T. (2020). *Guide to assembly language: a concise introduction*. Springer Nature.

NASM. (n.d.). Nasm. Retrieved February 19, 2022, from https://www.nasm.us/

HORSTR. (2020, May 17). *Difference between NASM, TASM, & MASM*. Stack Overflow. https://stackoverflow.com/questions/61857760/difference-between-nasm-tasm-masm

Pedamkar, P. (2021, March 30). *What is Assembly Language?* EDUCBA. Retrieved February 19, 2022, from https://www.educba.com/what-is-assembly-language/

Naikis, M. (2015, September 30). *Which programming language is used to write a BIOS program?* Software Engineering Stack Exchange. Retrieved February 19, 2022, from https://softwareengineering.stackexchange.com/questions/298628/which-programming-language-is-used-to-write-a-bios-program

Ghoshal, A. (2016). *Are assembly programming languages still in use? If so, what are their uses? Which organisations use them?* Quora. Retrieved February 19, 2022, from https://www.quora.com/Are-assembly-programming-languages-still-in-use-If-so-what-are-their-uses-Which-organisations-use-them

Fernando, J. (2020, December 20). *Assembly Language Definition*. Investopedia. Retrieved February 19, 2022, from https://www.investopedia.com/terms/a/assembly-language.asp

An Introduction to Real-Time Embedded Systems. (2019, February 19). Total Phase Blog. Retrieved February 19, 2022, from https://www.totalphase.com/blog/2019/12/an-introduction-to-real-time-embedded-systems/

Maxfield, C. (2007, August 7). *Is anyone still using assembly language? You betcha! (Part 1)*. Eetimes. Retrieved February 19, 2022, from https://www.eetimes.com/is-anyone-still-using-assembly-language-you-betcha-part-1/

A Beginner's Guide to Digital Signal Processing (DSP) | Design Center | Analog Devices. (n.d.). Analog. Retrieved February 19, 2022, from https://www.analog.com/en/design-center/landing-pages/001/beginners-guide-to-dsp.html

Heynthan, R., & Fadzlan, F. (2016). N ANALYSIS AND IMPLEMENTATION OF ASSEMBLY LANGUAGE PROGRAMMING BY USING TASM INCORPORATING WITH SECURITY CONCEPTS. *N ANALYSIS AND IMPLEMENTATION OF ASSEMBLY LANGUAGE PROGRAMMING BY USING TASM INCORPORATING WITH SECURITY CONCEPTS*, *2*(11), 163–173. http://ijirse.com/wp-content/upload/2016/02/161ijirse.pdf