TUNKU ABDUL RAHMAN UNIVERSITY COLLEGE
FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY

Bachelor of Science (Hons.) Management Mathematics with Computing

Year 2 Semester 3

BACS2093 (Group 1)

Practical Assignment

BACS2093 Operating Systems (AY 202101)

Name(Block Capital)	Registration No	Signature	Marks
1. TAN YIN YUEN	19WMR05960		
2. CHEW HWA ERN	19WMR04184		

Lecturer/Tutor's Name: Ms Chin Chai Lim

Task 1:

Screen Output(s)

```
yinyuen@yinyuen-VirtualBox:~$ ./menuscript


        COLLEGE MANAGEMENT MENU
=====================================

1 - Add New Students
2 - Add New Courses
3 - Grade Students (Enter marks for students)
4 - Search Student Details
5 - Search Course Details
6 - Search Student's Result

Q - QUIT PROGRAMME

Please Select A Choice  :
```

Figure 1.1 Upon launching the program. The main menu will show.

```
        COLLEGE MANAGEMENT MENU
=====================================

1 - Add New Students
2 - Add New Courses
3 - Grade Students (Enter marks for students)
4 - Search Student Details
5 - Search Course Details
6 - Search Student's Result

Q - QUIT

Please Select A Choice  : 1



        PROGRAMME SELECTIONS MENU
=====================================

T - RIT (Bachelor in Information Technology)
D - RSD (Bachelor in Software Development)
S - RST (Bachelor in Interactive Software Technology)
E - REI (Bachelor in Enterprise Information System)
F - RSF (Bachelor in Software Engineering)
O - Add Other Programme Code

Q - QUIT (Return to College Management Menu)

Please Select A Choice   :
```

Figure 1.2 Select 1 to Add a New Student

```
        COLLEGE MANAGEMENT MENU
=====================================

1 - Add New Students
2 - Add New Courses
3 - Grade Students (Enter marks for students)
4 - Search Student Details
5 - Search Course Details
6 - Search Student's Result

Q - QUIT

Please Select A Choice  : 2


**press [Q/q] to quit programme**


        ADD NEW COURSE FORM
=====================================

Course Code (eg. BACS2093)        :
```

Figure 1.3 Select 2 to Add a New Course

```
yinyuen@yinyuen-VirtualBox:~$ ./menuscript


        COLLEGE MANAGEMENT MENU
=====================================

1 - Add New Students
2 - Add New Courses
3 - Grade Students (Enter marks for students)
4 - Search Student Details
5 - Search Course Details
6 - Search Student's Result

Q - QUIT

Please Select A Choice  : 3


**press [Q/q] to quit programme**


                STUDENT VALIDATION FORM


        Enter Student ID (00XXX0000)    :
```

Figure 1.4 Select 3 to Enter Results for a Student.

```
        COLLEGE MANAGEMENT MENU
=====================================

1 - Add New Students
2 - Add New Courses
3 - Grade Students (Enter marks for students)
4 - Search Student Details
5 - Search Course Details
6 - Search Student's Result

Q - QUIT

Please Select A Choice  : 4


**press [Q/q] to quit programme**


        **please enter an existing student ID to conduct search**


                SEARCH STUDENT DETAILS FORM


        Enter Student ID        :
```

Figure 1.5 Select 4 to Search Student Details

```
        COLLEGE MANAGEMENT MENU
========================================

1 - Add New Students
2 - Add New Courses
3 - Grade Students (Enter marks for students)
4 - Search Student Details
5 - Search Course Details
6 - Search Student's Result

Q - QUIT

Please Select A Choice  : 5


**press [Q/q] to quit programme**


   **please enter an existing course code to conduct search**


                SEARCH COURSE DETAILS FORM


        Enter Course Code       :
```

Figure 1.6 Select 5 to Search Course

```
        COLLEGE MANAGEMENT MENU
========================================

1 - Add New Students
2 - Add New Courses
3 - Grade Students (Enter marks for students)
4 - Search Student Details
5 - Search Course Details
6 - Search Student's Result

Q - QUIT

Please Select A Choice  : 6


**press [Q/q] to quit programme**


   **please enter an existing student ID to conduct search**


                SEARCH RESULTS DETAILS FORM


        Enter Student ID (00XXX0000)    :
```

Figure 1.7 Enter 6 to Search Student Results.

```
        COLLEGE MANAGEMENT MENU
========================================

1 - Add New Students
2 - Add New Courses
3 - Grade Students (Enter marks for students)
4 - Search Student Details
5 - Search Course Details
6 - Search Student's Result

Q - QUIT PROGRAMME

Please Select A Choice  : q

Thanks for using this program!


Exiting the program...


yinyuen@yinyuen-VirtualBox:~$
```

Figure 1.8 If the user enters Q or q at any given input field, the program will exit or go back to the main menu.

```
yinyuen@yinyuen-VirtualBox:~$ ./menuscript


        COLLEGE MANAGEMENT MENU
========================================

1 - Add New Students
2 - Add New Courses
3 - Grade Students (Enter marks for students)
4 - Search Student Details
5 - Search Course Details
6 - Search Student's Result

Q - QUIT PROGRAMME

Please Select A Choice  : 7

You Have Entered an Invalid Input, Please Try Again. (1-6)

Please Select A Choice  :
```

Figure 1.9 If the user enters an invalid input, an error message will pop out. Then the user needs to re enter their choice again.

Sample Codes

```
#!/bin/bash

 echo
 echo
 echo -e "\e[1m\t\e[40;38;5;82mCOLLEGE MANAGEMENT MENU\e[0m"
 echo -e "\e[36m=====================================\e[0m"
 echo
 echo -e "\e[33m1 \e[0m- Add New Students"
 echo -e "\e[92m2 \e[0m- Add New Courses"
 echo -e "\e[93m3 \e[0m- Grade Students (Enter marks for students)"
 echo -e "\e[94m4 \e[0m- Search Student Details"
 echo -e "\e[95m5 \e[0m- Search Course Details"
 echo -e "\e[96m6 \e[0m- Search Student's Result"
 echo " "
 echo -e "\e[91mQ \e[0m- QUIT PROGRAMME"
 echo " "
 #The above printed the choice of selection.

continue=i
until [ "$continue" = "correct" ]
do
 echo -n -e "\e[2mPlease Select A Choice\t: "; read menuchoice
 echo

 case "$menuchoice" in
 1)
    ./addstudentscript
    echo " "
    continue=correct
    ;;
 2)
    ./addcoursescript
    echo " "
    continue=correct
    ;;
 3)
    ./gradescript
    echo " "
    continue=correct
    ;;
 4)
    ./searchstudentscript
    echo " "
    continue=correct
    ;;
 5)
    ./searchcoursescript
    echo " "
    continue=correct
    ;;
 6)
    ./searchresultscript
```

```
    echo " "
    continue=correct
    ;;
  [Qq])
    continue=correct


    echo -e "\e[1;40;38mThanks for using this program!\e[0m"
    echo " "
    echo " "
    echo "Exiting the program..."
    echo " "

    #for i in  {16..21} {21..16} ;
    # do
    # echo -en "\e[48;5;${i}m \e[0m" ;
    #done

    echo " "
    sleep 2
    exit
    ;;
  *)
    echo "You Have Entered an Invalid Input, Please Try Again. (1-6)"
    echo " "
    ;;
  esac

done
```

Description:

#It will loop until the user selects a correct input.
#Each selection of input runs different scripts or to exit the program.
#If there is any invalid input, an error message will pop out and prompt the user to reenter their choices again.

Task 2:

Screen Output(s) - Add New Student

```
        PROGRAMME SELECTIONS MENU
====================================

T - RIT (Bachelor in Information Technology)
D - RSD (Bachelor in Software Development)
S - RST (Bachelor in Interactive Software Technology)
E - REI (Bachelor in Enterprise Information System)
F - RSF (Bachelor in Software Engineering)
O - Add Other Programme Code

Q - QUIT (Return to College Management Menu)

Please Select A Choice   : 0


You Have Entered an Invalid Option!
Please Try Again.

Please Select A Choice   : O

Please Enter Your Programme Code (RXX) or [Q/q] to return to Programme Selection Menu :
RMM

**press [Q/q] to quit programme**


        ADD NEW STUDENT FORM
====================================

Student ID (19XXX0000)                          : 19WMR04184
Student Name (Full Name)                        : CHEW HWA ERN
Birth Date (YYYY-MM-DD)                         : 1999-08-12
Contact Number (000-0000000)                    : 017-2099778
Mailing Address                                 : SEGAMBUT
Email Address (name-xx00@student.tarc.edu.my)   : chewhe-wm19@student.tarc.edu.my

Do You Want To Save This Student's Information? YES[Y] NO[N]    :Y

RECORD SAVED.

Do You Want To Add Another Student? YES[Y] NO[N]                :
```

FIgure 2.1 A basic flow of users to add new student information. The user needs to choose the programme first, then enter their information based on the given format. The programme will prompt a confirmation message before saving the input information. If yes, their data will be saved in the text file as shown as below. Choosing No will bring the user back to the main menu.

```
Open    student.txt    Save
1 RMM:19WMR05960:Tan Yin Yuen:2000-8-26:014-6611848:37 Lorong Serunai 3:tanyy-wp18@student.tarc.edu.my
2 RIT:19WMR01017:Choy MX:2000-8-26:011-61466268:5 Lorong Masria 3:choymx-wp18@student.tarc.edu.my
3 RMM:19WMR04184:CHEW HWA ERN:1999-08-12:017-2099778:SEGAMBUT:chewhe-wm19@student.tarc.edu.my
```

Figure 2.2 The records will then be saved in the student text file.
Users can choose to continue to adding new student's information or exit to the main menu.

```
        ADD NEW STUDENT FORM
=====================================

Student ID (19XXX0000)                          : 19WMR04184
THIS ID ALREADY EXISTS. PLEASE TRY AGAIN.
Student ID (19XXX0000)                          :
```

Figure 2.3 Duplicated Student ID will prompt the user to enter another ID that does not exist already.

Sample Codes (Add New Student)

```
#!/bin/bash
#PROGRAMME SELECTIONS MENU

echo
echo
echo -e "\e[1m\t\e[30;48;5;82mPROGRAMME SELECTIONS MENU\e[0m"
echo -e "\e[36m=====================================\e[0m"
echo " "
echo -e "\e[33mT \e[0m- RIT (Bachelor in Information Technology)"
echo -e "\e[92mD \e[0m- RSD (Bachelor in Software Development)"
echo -e "\e[93mS \e[0m- RST (Bachelor in Interactive Software Technology)"
echo -e "\e[94mE \e[0m- REI (Bachelor in Enterprise Information System)"
echo -e "\e[95mF \e[0m- RSF (Bachelor in Software Engineering)"
echo -e "\e[96mO \e[0m- Add Other Programme Code"
echo " "
echo -e "\e[91mQ \e[0m- QUIT (Return to College Management Menu)"
echo " "

studentchoice=i
until [ "$studentchoice" = "correct" ]
do
echo -n -e "\e[2mPlease Select A Choice\t \e[0m: "; read studentchoice
echo

case "$studentchoice" in
[TtDdSsEeFfOo])
    case "$studentchoice" in
    [Tt])
          programme="RIT"
          studentchoice=correct
          ;;
    [Dd])
          programme="RSD"
          studentchoice=correct
          ;;
    [Ss])
          programme="RST"
          studentchoice=correct
```

```
          ;;
    [Ee])
          programme="REI"
          studentchoice=correct
          ;;
    [Ff])
          programme="RSF"
          studentchoice=correct
          ;;
    [Oo])
          again=i

          until [ "$again" = "ok" ]
          do
          echo "Please Enter Your Programme Code (RXX) or [Q/q] to return to
Programme Selection Menu :"; read other

          if [[ $other =~ ^[A-Z]{3}$ ]] || [[ "$other" = "Q" ]] || [[
"$other" = "q" ]]; then

            if [[ $other =~ ^[A-Z]{3}$ ]]; then
             programme="$other"
             again=ok
            else
             again=ok
             clear
             ./addstudentscript
             exit
            fi

          else
           echo "SELECTION NOT FOUND. PLEASE TRY AGAIN"
          fi
          done

          studentchoice=correct
          ;;
    esac

#ADD STUDENT FORM

     echo
     echo -e "\e[2m**press [Q/q] to quit programme**\e[0m"
     echo
     echo
     echo -e "\e[40;38;5;82m\tADD NEW STUDENT FORM\e[0m"
     echo -e "\e[94m=====================================\e[0m"
     echo

     stuid=i
     until [ "$stuid" = "ok" ]
     do
      echo -en "Student ID (19XXX0000)\t\t\t\t: "; read studentid
      if [[ $studentid =~ ^[0-9]{2}[A-Z]{3}[0-9]{5}$ ]] || [[ $studentid =~
```

```
^[Qq]$ ]]; then

      if [[ $studentid =~ ^[0-9]{2}[A-Z]{3}[0-9]{5}$ ]]; then
      check="$(grep "$studentid" "student.txt")"

       while IFS=: read -r prog sid remainingdetails
         do
           if ! [ "$studentid" = "$sid" ]; then
           stuid=ok
           id=$studentid
           else
           echo -e "\e[31mTHIS ID ALREADY EXISTS. PLEASE TRY AGAIN.\e[0m"
           fi
           break
       done <<< $check

      else
       echo
       echo "RETURNING TO PROGRAMME SELECTION MENU...THIS MIGHT TAKE A
SECOND"
       sleep 2
       stuid=ok
       clear
       ./addstudentscript
       exit
      fi

     else
     echo -e "\e[31mINVALID STUDENT ID. PLEASE TRY AGAIN.
(00XXX00000)\t\e[0m"
     fi
    done

    stuname=i
    until [ "$stuname" = "ok" ]
    do
    echo -en "Student Name (Full Name)\t\t\t: "; read name
    if [[ $name =~ ^[*\ A-Za-z]+$ ]] || [[ $name =~ ^[Qq]$ ]]; then

      if [[ $name =~ ^[Qq]$ ]]; then
       echo " "
       echo "RETURNING TO PROGRAMME SELECTION MENU...THIS MIGHT TAKE A
SECOND"
       sleep 2
       stuname=ok
       clear
       ./addstudentscript
       exit
      else
       stuname=ok
      fi

   else
    echo -e "\e[31mINVALID NAME. PLEASE TRY AGAIN. (FULL NAME AS PER IC)
```

```
\e[0m"
    fi
    done

     birth=i
     until [ "$birth" = "ok" ]
     do
     echo -en "Birth Date (YYYY-MM-DD)\t\t\t\t: "; read date
     if [[ $date =~ ^[Qq]$ ]] || [[ $date =~
^[0-9]{4}-[0-9]?[0-9]-[0-9]?[0-9]$ ]] && date -d "$date" >/dev/null 2>&1;
then

       if [[ $date =~ ^[0-9]{4}-[0-9]?[0-9]-[0-9]?[0-9]$ ]] && date -d
"$date" >/dev/null 2>&1; then
        birth=ok
       else
        echo
        echo "RETURNING TO PROGRAMME SELECTION MENU...THIS MIGHT TAKE A
SECOND"
        sleep 2
        birth=ok
        clear
        ./addstudentscript
        exit
       fi

     else
     echo -en "\n\e[31mINCORRECT DATE FORMAT. PLEASE TRY AGAIN.\e[0m\n"
     fi
     done

     con=i
     until [ "$con" = "ok" ]
     do
     echo -en "Contact Number (000-0000000)\t\t\t: "; read contact
     if [[ $contact =~ ^[0-9]{3}-[0-9]{7}[0-9]?$ ]] || [[ $contact =~ ^[Qq]$
]]; then

       if [[ $contact =~ ^[0-9]{3}-[0-9]{7}[0-9]?$ ]]; then
        con=ok
       else
        echo
        echo "RETURNING TO PROGRAMME SELECTION MENU...THIS MIGHT TAKE A
SECOND"
        sleep 2
        con=ok
        clear
        ./addstudentscript
        exit
       fi

     else
      echo -en "\n\e[31mINVALID CONTACT NUMBER. PLEASE TRY AGAIN.\e[0m"
     fi
```

```
     done

     add=i
     until [ "$add" = "ok" ]
     do
     echo -en "Mailing Address\t\t\t\t\t: "; read address
     if [ -n "$address" ] || [[ $address =~ ^[Qq]$ ]]; then

       if [ -n "$address" ]; then
        add=ok
       else
        echo
        echo "RETURNING TO PROGRAMME SELECTION MENU...THIS MIGHT TAKE A
SECOND"
        sleep 2
        add=ok
        clear
        ./addstudentscript
        exit
       fi

     else
     echo -en "\n\e[31mEMPTY MAILING ADDRESS. PLEASE TRY AGAIN.\e[0m"
     fi
     done

     mailing=i
     until [ "$mailing" = "ok" ]
     do
     echo -en "Email Address (name-xx00@student.tarc.edu.my)\t: "; read mail
     if [[ $mail =~ ^[A-Za-z]+-[A-Za-z]{2}[0-9]{2}@student.tarc.edu.my$ ]]
|| [[ $mail =~ ^[Qq]$ ]]; then

       if [[ $mail =~ ^[A-Za-z]+-[A-Za-z]{2}[0-9]{2}@student.tarc.edu.my$
]]; then
        mailing=ok
       else
        echo
        echo "RETURNING TO PROGRAMME SELECTION MENU...THIS MIGHT TAKE A
SECOND"
        sleep 2
        mailing=ok
        clear
        ./addstudentscript
        exit
       fi

     else
      echo -en "\n\e[31mTHIS EMAIL IS NOT REGISTERED UNDER THE TARC DOMAIN.
PLEASE TRY AGAIN.\e[0m"
     fi
     done

     save=i
```

```
     until [ "$save" = "ok" ]
     do
     echo
     echo -en "\e[7mDo You Want To Save This Student's Information? YES[Y]
NO[N]\t\e[0m:"; read tosave

     case "$tosave" in
     [Yy])
          echo " "
          echo -en "\e[0m\e[5m\tSaving New Student Record...\e[0m"
          sleep 3
          echo -en "\r\e[0KRECORD SAVED."
          echo "$programme:$id:$name:$date:$contact:$address:$mail" >>
student.txt
          save=ok
          echo
          ;;
     [Nn])
          echo
          echo "Going Back to Add New Student Form..."
          sleep 2
          save=ok
          clear
          ./addstudentscript
          exit
          ;;
     *)
          echo -en "\n\e[31mINVALID OPTION.PLEASE TRY AGAIN.\e[0m"
          ;;
     esac
     done

     new=i
     until [ "$new" = "correct" ]
     do
     echo
     echo -en "\e[7mDo You Want To Add Another Student? YES[Y]
NO[N]\t\t\e[0m:"; read addnew
     case "$addnew" in
          [Yy])
                new=correct
                clear
                ./addstudentscript
                exit
                ;;
          [Nn])
                new=correct
                clear
                ./menuscript
                exit
                ;;
          *)
                echo -en "\n\e[31mINVALID OPTION.PLEASE TRY AGAIN.\e[0m"
                ;;
```

```
     esac
     done
    echo " "
    ;;

[Qq])
    echo "RETURNING TO MAIN MENU..."
    sleep 2
    clear
    ./menuscript
    exit
    ;;

*)
    echo
    echo - en "\n\e[31mINVALID OPTION. PLEASE TRY AGAIN.\e[0m"
    echo
    ;;
esac
done
```

Description:
#Users are prompted to choose a programme into which a new student is to be added.
#The user may select O or o if the listed programmes are not their choice. Only 3 Letters Programme input will be accepted. (i.e RMM)
#The user will then need to input ID, Name, Birth Date, Contact Number, Mailing Address and Email Address by their respective format, failing which requires users to re-enter.
#If users have confirmed to save the details, the information will be saved in a text file, else the program will loop back to the beginning of the form to add new student information.
#After users confirm to save their details, the program will ask the user if they wish to continue adding new students or exit to the main menu.
#Duplicated IDs are not allowed as IDs should be unique.

Screen Output(s) - Search Student

```
                SEARCH STUDENT DETAILS FORM

        Enter Student ID        :       19WMR04184
------------------------------------------------------------------------

Student Name            : CHEW HWA ERN

Contact Number          : 0172099778

Mailing Address         : KUALA LUMPUR

Email                   : chewhe-wm19@student.tarc.edu.my

Search Another Student? YES[Y] OR NO[N] :
```

Figure 2.4 To Search Student Details

```
**press [Q/q] to quit programme**

        **please enter an existing student ID to conduct search**

                SEARCH STUDENT DETAILS FORM

        Enter Student ID        :       19wmr05960
------------------------------------------------------------------------

STUDENT NOT FOUND. PLEASE TRY AGAIN BY ENTERING A VALID ID. (00XXX00000)

        Enter Student ID        :
```

Figure 2.5 Only Saved Student Details and Correct Format ID will show a result. ID entered should be in Capital letters

Sample Code (Search Student)

```
#!/bin/bash

echo
echo -e "\e[2m**press [Q/q] to quit programme**\e[0m"
echo
echo
echo -e "\e[90m  **please enter an existing student ID to conduct
search**\e[0m"
echo
echo
echo -e "\e[40;38;5;82m\t\tSEARCH STUDENT DETAILS FORM\e[0m"
echo
echo

loop=y
until [ "$loop" = "n" ]
do
 echo -en "\e[2m    Enter Student ID :     \e[0m"; read inputcode
 echo -e
"\e[94m_____________________________________________________________________________
_______\e[0m"

 sleep 1

 if [[ "$inputcode" =~ ^[Qq]$ ]]; then
 loop=n
 echo
 echo "RETURNING TO MAIN MENU..."
 sleep 2
 clear
 ./menuscript
 exit

 else
 search="$(grep "$inputcode" "student.txt")"

 while IFS=: read -r programme id name date number address mail
  do
    if [ "$inputcode" == "$id" ];then
      echo
      echo -e "Student Name\t\t: $name"
      echo
      echo -e "Contact Number\t\t: $number"
      echo
      echo -e "Mailing Address\t\t: $address"
      echo
      echo -e "Email\t\t\t: $mail"
      echo
      loop=n
    else
      echo
      echo -e "\e[31mSTUDENT NOT FOUND. PLEASE TRY AGAIN BY ENTERING A VALID
```

```
ID. (00XXX00000) \e[0m"
      echo
    fi
  done <<< $search

fi
done

 until [ "$quit" = "y" ]
 do
  echo
  echo -en "\e[7mSearch Another Student? YES[Y] OR NO[N]\t\e[0m: ";read
choice1
  echo

   case "$choice1" in
    [Yy])
         quit=y
         clear
         ./searchstudentscript
   exit
         ;;
    [Nn])
         quit=y
         echo
         echo "RETURNING TO MAIN MENU..."
         echo
         sleep 2
         clear
         ./menuscript
         exit
         ;;
    *)
         echo -en "\n\e[31mINVALID OPTION.PLEASE TRY AGAIN.\e[0m"
   echo " "
         ;;
    esac
    done
   echo ""
 done
```

Descriptions:
#After the user enters the student ID to be searched, the programme automatically displays the student’s information retrieved from the student text file.
#Users can only successfully retrieve student’s information by keying in an existing student’s ID in the correct format.

Task 3:

Screen Output (s) - Add New Course

```
        ADD NEW COURSE FORM
=====================================

Course Code (eg. BACS2093)      : BACS2093
Course Name                     : OS
Credit Hours (1-12)             : 4

Do You Want To Save This Course's Information? YES[Y] NO[N]     : y

RECORD SAVED.
Do You Want To Add Another Course? YES[Y] NO[N]        :
```

FIgure 3.1 Input the information of the new course with their respective format.
The data will be saved in a text file as shown below.

```
addcoursescript                    course.txt
1 BACS2093:OS:4
2 BAMS2054:AS:4
3 BAMS2044:MS:4
```

Figure 3.2 Saved Course Information

```
        ADD NEW COURSE FORM
=====================================

Course Code (eg. BACS2093)        : BACS2093
THIS COURSE CODE ALREADY EXISTS. PLEASE TRY AGAIN.
Course Code (eg. BACS2093)        : BAMS2054
THIS COURSE CODE ALREADY EXISTS. PLEASE TRY AGAIN.
Course Code (eg. BACS2093)        : BAMS2044
Course Name                     : AS
Credit Hours (1-12)             : 4

Do You Want To Save This Course's Information? YES[Y] NO[N]     : y

RECORD SAVED.
Do You Want To Add Another Course? YES[Y] NO[N]        : n
```

Figure 3.3 Duplicated Course ID is not allowed.

Sample Codes (Add New Course)

```
#!/bin/bash

echo
echo -e "\e[2m**press [Q/q] to quit programme**\e[0m"
echo
echo
echo -e "\e[40;38;5;82m\tADD NEW COURSE FORM\e[0m"
echo -e "\e[94m=====================================\e[0m"
echo

course=i
until [ "$course" = "ok" ]
do
echo -n "Course Code (eg. BACS2093)        : "; read coursecode
if [[ $coursecode =~ ^[A-Z]{4}[0-9]{4}$ ]] || [[ $coursecode =~ ^[Qq]$ ]];
then
  if [[ $coursecode =~ ^[A-Z]{4}[0-9]{4}$ ]]; then
       check="$(grep "$coursecode" "course.txt")"
       while IFS=: read -r code remainingdetails
          do
            if ! [ "$coursecode" = "$code" ]; then
            course=ok
            else
            echo -e "\e[31mTHIS COURSE CODE ALREADY EXISTS. PLEASE TRY
AGAIN.\e[0m"
            fi
            break
       done <<< $check

  else
   echo
   echo "RETURNING TO MAIN MENU..."
   sleep 2
   course=ok
   clear
   ./menuscript
   exit
  fi

else
 echo
 echo -e "\e[31mINVALID COURSE CODE. PLEASE TRY AGAIN. (XXXX0000)\e[0m"
fi
done

coursename=i
until [ "$coursename" = "ok" ]
do
echo -en "Course Name\t\t\t: "; read name

if [[ $name =~ ^[*\ A-Za-z]+[-]*[*\ A-Za-z]+$ ]] || [[ $name =~ ^[Qq]$ ]];
then
```

```
  if [[ $name =~ ^[*\ A-Za-z]+[-]*[*\ A-Za-z]+$ ]]; then
    coursename=ok
  else
    echo "RETURNING TO MAIN MENU..."
    sleep 2
    coursename=ok
    clear
    ./menuscript
    exit
  fi

else
  echo
  echo -e "\e[31mINVALID INPUT. PLEASE TRY AGAIN. (ENTER FULL COURSE
NAME)\e[0m"
  fi
done


proceed=i
until [ "$proceed" = can ]
do
  echo -en "Credit Hours (1-12)\t\t: "; read hour
  if [[ $hour =~ ^[1]?[0-9]$ ]] && [ "$hour" -ge 1 -a "$hour" -le 12 ] || [[
$hour =~ ^[Qq]$ ]]; then
    if [[ $hour =~ ^[1]?[0-9]$ ]] && [ "$hour" -ge 1 -a "$hour" -le 12 ];
then
    proceed=can
    else
    echo
    echo "RETURNING TO MAIN MENU..."
    sleep 2
    proceed=can
    clear
    ./menuscript
    exit
    fi
  else
    echo
    echo -e "\e[31mHOURS NOT WITHIN RANGE. PLEASE TRY AGAIN.\e[0m"
  fi
done

echo " "

save=i
until [ "$save" = "ok" ]
  do
  echo -en "\e[7mDo You Want To Save This Course's Information? YES[Y]
NO[N]\t\e[0m: "; read tosave

    case "$tosave" in
    [Yy])
          echo
```

```
          echo -en "\e[0m\e[5m\tSaving New Course Record...\e[0m"
          sleep 3
          echo -en "\r\e[0KRECORD SAVED."
          echo "$coursecode:$name:$hour" >> course.txt
          save=ok
          echo
          ;;
    [Nn])
          echo "RETURNING TO MAIN MENU..."
          echo
          sleep 2
          save=ok
          clear
          ./menuscript
          exit
          ;;
    *)
          echo -en "\n\e[31mINVALID OPTION.PLEASE TRY AGAIN.\e[0m"
          echo
          ;;
    esac
  done

add=i
until [ "$add" = "correct" ]
do
 echo -en "\e[7mDo You Want To Add Another Course? YES[Y] NO[N] \t\e[0m: ";
read addcourse

 case "$addcourse" in
 [Yy])
    add=correct
    clear
    ./addcoursescript
    exit
    ;;
 [Nn])
    add=correct
    clear
    ./menuscript
    exit
    ;;
 *)
    echo -en "\n\e[31mINVALID OPTION.PLEASE TRY AGAIN.\e[0m"
    echo " "
    ;;
 esac
done
#The new added course should be unique and does not duplicate with the saved
course.
#The course code, course name and credit hour have their own format and
validation while user inputting the new course.
```

Screen Output (s) - Search Course

```
**press [Q/q] to quit programme**

   **please enter an existing course code to conduct search**

                SEARCH COURSE DETAILS FORM

        Enter Course Code       :       BACS2093
--------------------------------------------------------------------------------
Course Name     : OS
Credit Hours    : 4

Search Another Course? YES[Y] OR NO[N]  : y
```

Figure 3.4 Search a Course by entering the course code retrieves the course name and credit hours.

```
   **please enter an existing course code to conduct search**

                SEARCH COURSE DETAILS FORM

        Enter Course Code       :       BMCS2113
--------------------------------------------------------------------------------
COURSE NOT FOUND. PLEASE TRY AGAIN BY ENTERING A VALID COURSE CODE. (XXXX0000)

        Enter Course Code       :
```

Figure 3.5 Only Saved Course and Correct Course Code Format will Show Results. Course code must only be in capital letters.

Sample Codes (Search Course)

```
#!/bin/bash
echo
echo -e "\e[2m**press [Q/q] to quit programme**\e[0m"
echo
echo
echo -e "   \e[90m**please enter an existing course code to conduct
search**\e[0m"
echo
echo
echo -e "\e[40;38;5;82m\t\tSEARCH COURSE DETAILS FORM\e[0m"
echo
echo

loop=y
until [ "$loop" = "n" ]
do
   echo -en "\e[2m  Enter Course Code:     \e[0m"; read inputcode
   echo -e
"\e[94m_____________________________________________________________
_______\e[0m"

   sleep 1

   if [[ "$inputcode" =~ ^[Qq]$ ]]; then
    loop=n
    echo "RETURNING TO MAIN MENU..."
    sleep 2
    clear
    ./menuscript
    exit
   else
    search="$(grep "$inputcode" "course.txt")"

    while IFS=: read -r code name hour
    do
     if [ "$inputcode" = "$code" ];then
      echo
      echo "Course Name   : $name"
      echo "Credit Hours  : $hour"
      echo
      loop=n
     else
      echo
      echo -e "\e[31mCOURSE NOT FOUND. PLEASE TRY AGAIN BY ENTERING A VALID
COURSE CODE. (XXXX0000) \e[0m"
      echo
     fi
   done <<< $search
      fi
done

 quit=n
```

```
 until [ "$quit" = "y" ]
 do
  echo
  echo -en "\e[7mSearch Another Course? YES[Y] OR NO[N]\t\e[0m: ";read
choice1

   case "$choice1" in
    [Yy])
         quit=y
         clear
   ./searchcoursescript
         exit
         ;;
    [Nn])
         quit=y
         echo "RETURNING TO MAIN MENU..."
         sleep 2
         clear
         ./menuscript
         exit
         ;;
    *)
         echo -en "\n\e[31mINVALID OPTION.PLEASE TRY AGAIN.\e[0m"
   echo
   echo
         ;;
    esac
   echo ""
 done
```

Description:

#After entering the existing course code in the correct format that starts with 4 capital letters followed by 4 numbers, the programme will automatically display the remaining details of the searched course i.e Course Name and Credit hours.

Task 4:

Screen Output(s)

```
        COLLEGE MANAGEMENT MENU
====================================

1 - Add New Students
2 - Add New Courses
3 - Grade Students (Enter marks for students)
4 - Search Student Details
5 - Search Course Details
6 - Search Student's Result

Q - QUIT

Please Select A Choice  : 3


**press [Q/q] to quit programme**


                STUDENT VALIDATION FORM


        Enter Student ID (00XXX0000)    :       19WMR04184
____________________________________________________________

Student Name            : CHEW HWA ERN

Enrolled Programme      : RMM

Do You Wish To Proceed with This Student? YES[Y] NO[N]  : Y
```

Figure 4.1 Upon loading option 3 from College Management Menu, users will be prompted with the Student Validation Form. Users can filter out the Student Name and Enrolled Programme of the student by entering the Student ID. The system will read from both the student file and course file to retrieve the student's details by the ID.

```
This student already has an existing record, any changes in grading will replace the old existing data.



Do You Wish To Overwrite Existing Semester Grades? YES[Y] NO[N] : Y

**press [Q/q] to exit form**
```

Figure 4.2 If the student already has an existing file record of semester grades, the programme overwrites the old results data with newly key-ed in data.

```
                STUDENT'S EXAMINATION MARKS FORM
_____________________________________________________________________


Academic Year (YYYY)            : 2020
Semester (1/2/3)                : 1
Course Code (eg. BACS2093)      : BACS2093
Marks Obtained (0-100)          : 88


Do You Wish To Continue Adding Marks To This Student's File? YES[Y] NO[N] : Y

Course Code (eg. BACS2093)      : BAMS2054
Marks Obtained (0-100)          : 88


Do You Wish To Continue Adding Marks To This Student's File? YES[Y] NO[N] : Y

Course Code (eg. BACS2093)      : BAMS2044
Marks Obtained (0-100)          : 88


Do You Wish To Continue Adding Marks To This Student's File? YES[Y] NO[N] : N



RECORD SAVED.

RETURNING TO STUDENT VALIDATION FORM...
```

Figure 4.3 When users confirms to add grade marks to the respective student's profile, they will be prompted with the Student's Examination Marks Form.
Users have to provide semester grades of different subjects at once in order to save the record into a text file altogether.

```
Open                                    19WMR04184.txt
Student ID      : 19WMR04184
Student Name    : CHEW HWA ERN
Academic Year   : 2021
Semester        : 1

==================================================================================================================
        Course Code             Marks Obtained              Grade Obtained              Remark                  Quality Point
==================================================================================================================
        BACS2093                        88                          A                   Excellent               16.0000
        BAMS2054                        88                          A                   Excellent               16.0000
                                                                                                            =========
                                                                                        Total Quality Point:    32.0000
```

Figure 4.4 Once Records have been saved, a text file will be named after the student's ID and the text file will contain the updated or key-ed in marks of the student in the said text file.

```
        COLLEGE MANAGEMENT MENU
=====================================

1 - Add New Students
2 - Add New Courses
3 - Grade Students (Enter marks for students)
4 - Search Student Details
5 - Search Course Details
6 - Search Student's Result

Q - QUIT

Please Select A Choice  : 6


**press [Q/q] to quit programme**


   **please enter an existing student ID to conduct search**


                SEARCH RESULTS DETAILS FORM


        Enter Student ID (00XXX0000)    :       19WMR04184
```

Figure 4.5 Users can search and print result details in the programme as well by entering the student ID.

```
                SEARCH RESULTS DETAILS FORM

        Enter Student ID (00XXX0000)    :       19WMR04184

Student ID      : 19WMR04184
Student Name    : CHEW HWA ERN
Academic Year   : 2020
Semester        : 1

==================================================================================================
        Course Code             Marks Obtained                  Grade Obtained                  Remark                  Quality Point
==================================================================================================
        BACS2093                        88                              A                       Excellent               16.0000
        BAMS2054                        88                              A                       Excellent               16.0000
        BAMS2044                        88                              A                       Excellent               16.0000
                                                                                                                        =========
                                                                                                Total Quality Point:    48.0000

Search Another Result? YES[Y] OR NO[N]  :
```

Figure 4.6 The programme displays content of the student's result text file.

```
Q - QUIT

Please Select A Choice  : Q

Thanks for using this program!


EXITING...
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

Figure 5 Quit programme.

Sample Codes (Marks Script)

```
#!/bin/bash

    if [ "$marks" -ge 80 -a "$marks" -le 100 ]; then
    grade=A
    GPA=40000
    Remarks=Excellent
    proceed=cont

    elif [ "$marks" -ge 75 -a "$marks" -le 79 ]; then
    grade=A-
    GPA=37500
    Remarks=Excellent
    proceed=cont

    elif [ "$marks" -ge 70 -a "$marks" -le 74 ]; then
    grade=B+
    GPA=35000
    Remarks=Good
    proceed=cont

    elif [ "$marks" -ge 65 -a "$marks" -le 69 ]; then
    grade=B
    GPA=30000
    Remarks=Good
    proceed=cont

    elif [ "$marks" -ge 60 -a "$marks" -le 64 ]; then
    grade=B-
    GPA=27500
    Remarks=Pass
    proceed=cont

    elif [ "$marks" -ge 55 -a "$marks" -le 59 ]; then
    grade=C+
    GPA=25000
    Remarks=Pass
    proceed=cont

    elif [ "$marks" -ge 50 -a "$marks" -le 54 ]; then
    grade=C
    GPA=20000
    Remarks=Pass
    proceed=cont

    elif [ "$marks" -ge 0 -a "$marks" -le 49 ]; then
    grade=F
    GPA=0
    Remarks=Failed
    proceed=cont

    else
     echo
```

```
      echo -en "\n\e[31mINPUT NOT IN RANGE.PLEASE TRY AGAIN. (0-100) \e[0m"
      echo

     fi

#This script stores the assignment of marks and grades.
```

Sample Codes (Grade Script)

```
#!/bin/bash

echo
echo -e "\e[2m**press [Q/q] to quit programme**\e[0m"
echo
echo
echo -e "\e[40;38;5;82m\t\tSTUDENT VALIDATION FORM\e[0m"
echo
echo


stuid=i
 until [ "$stuid" = "ok" ]
 do
  echo -en "\e[2m   Enter Student ID (00XXX0000) :     \e[0m"; read
studentid
  echo -en
"\e[94m_____________________________________________________________________
_______\e[0m"
  echo

 if  [[ $studentid =~ ^[Qq]$ ]]; then
   echo
   echo "RETURNING TO MAIN MENU..."
   echo
   sleep 2
   stuid=ok
   clear
   ./menuscript
   exit

 else
   check="$(grep "$studentid" "student.txt")"
   while IFS=: read -r programme id name others
    do
      if [ "$studentid" = "$id" ]; then
       echo " "
       echo -e "Student Name\t\t: $name"
       echo
       echo -e "Enrolled Programme\t: $programme"
       echo
       studentname=$name
       stuid=ok
      else
       echo
       echo -e "\e[31mNO RECORD FOUND. PLEASE ENTER ANOTHER STUDENT ID.
(00XXX00000) \e[0m"
       echo
      fi
   done <<< $check

 fi
```

```
 done

continue=i
until [ "$continue" = "ok" ]
do
 echo -en "\e[7mDo You Wish To Proceed with This Student? YES[Y]
NO[N]\t\e[0m: "; read tosave

  case "$tosave" in
  [Yy])
   continue=ok
   ;;
  [Nn])
   echo
   echo "RETURNING TO STUDENT VALIDATION FORM...THIS MIGHT TAKE A SECOND"
   echo
   sleep 2
   continue=ok
   clear
   ./gradescript
   exit
   ;;
  *)
   echo -en "\n\e[31mINVALID OPTION.PLEASE TRY AGAIN. YES[Y] OR NO[N]\e[0m"
   echo
   ;;
  esac
done

   if [ -f "$studentid".txt ] && [ -s "$studentid".txt ]; then
    echo " "
    echo " "
    echo " "
    echo " "
    echo " "
    echo -en "\n\t\t\e[104mThis student already has an existing record, any
changes in grading will replace the old existing data.\e[0m"
    echo " "
    echo " "
    echo " "
    echo " "
    echo " "

   over=i
   until [ "$over" = "ok" ]
   do
    echo
    echo -en "\e[7mDo You Wish To Overwrite Existing Semester Grades?
YES[Y] NO[N] \e[0m: "; read overwrite

    case "$overwrite" in
    [Yy])
         rm "$studentid".txt
         over=ok
```

```
          ;;
     [Nn])
          echo
          echo "RETURNING TO STUDENT VALIDATION FORM..."
          echo
          sleep 2
          over=ok
          clear
          ./gradescript
          exit
          ;;
     *)
          echo " "
          echo -en "\n\e[31mINVALID OPTION.PLEASE TRY AGAIN.\e[0m"
          echo " "
          ;;

     esac
    done
    fi

#STUDENT'S EXAMINATION MARKS FORM

echo
echo -e "\e[2m**press [Q/q] to exit form**\e[0m"
echo
echo
echo -e "\e[40;38;5;82m\t\tSTUDENT'S EXAMINATION MARKS FORM\e[0m"
echo -en
"\e[94m_____________________________________________________________________________
_______\e[0m"
echo
echo

academic=i

until [ "$academic" = "ok" ]
do
 echo " "
 echo -en "\e[2mAcademic Year (YYYY)\t\t: "; read year

  if [[ $year =~ ^[0-9]{4}$ ]] || [[ $year =~ ^[Qq]$ ]]; then

   if [[ $year =~ ^[0-9]{4}$ ]]; then
    YEAR=$year
    academic=ok
   else
    echo
    echo "RETURNING TO STUDENT VALIDATION FORM..."
    echo
    sleep 2
    academic=ok
    clear
    ./gradescript
```

```
    exit
    fi
   else
    echo
    echo -e "\e[31mINVALID ACADEMIC YEAR. PLEASE TRY AGAIN.\e[0m"
    echo
   fi
done


    sem=i

    until [ "$sem" = "ok" ]
    do
     echo -en "Semester (1/2/3)\t\t: "; read semester

      if [[ $semester =~ ^[1-3]$ ]] || [[ $semester =~ ^[Qq]$ ]]; then

          if [[ $semester =~ ^[1-3]$ ]]; then
           SEMESTER=$semester
           sem=ok
          else
           echo
           echo "RETURNING TO STUDENT VALIDATION FORM..."
           echo
           sleep 2
           sem=ok
           clear
           ./gradescript
           exit
          fi
      else
       echo
       echo -e "\e[31mINVALID SEMESTER. PLEASE TRY AGAIN. (1/2/3)\e[0m"
       echo
      fi
    done

    echo -e "Student ID\t: $studentid" >> $studentid.txt
    echo -e "Student Name\t: $studentname" >> $studentid.txt
    echo -e "Academic Year\t: $YEAR" >> $studentid.txt
    echo -e "Semester\t: $SEMESTER" >> $studentid.txt
    echo " " >> $studentid.txt
    echo
"==========================================================================
=============================================================" >>
$studentid.txt
    echo -e "\tCourse Code\t\tMarks Obtained\t\t\tGrade
Obtained\t\t\tRemark\t\t\tQuality Point\t" >> $studentid.txt
    echo
"==========================================================================
=============================================================" >>
$studentid.txt
```

```
    total=0
    next=i
    until [ "$next" = "ok" ]
    do

    newcourse=i
    until [ "$newcourse" = "ok" ]
    do
     course=i
     until [ "$course" = "ok" ]
     do
      echo -en "\e[2mCourse Code (eg. BACS2093)\t: "; read coursecode
      if [[ $coursecode =~ ^[Qq]$ ]]; then
          echo
          echo "RETURNING TO STUDENT VALIDATION FORM..."
          echo
          sleep 2
          course=ok
          clear
          ./gradescript
          exit
      else
       check="$(grep "$coursecode" "course.txt")"

       while IFS=: read -r code name hour
          do
            if [ "$coursecode" = "$code" ]; then
             credithour=$hour
             savedcourse=$code
             course=ok
             break
            else
             echo
             echo -e "\e[31mTHIS COURSE DOES NOT EXIST. PLEASE TRY
AGAIN.\e[0m"
             echo
            fi
          done <<< $check
     fi
     done

    search="$(grep "$savedcourse" "$studentid.txt")"
     if [ "$savedcourse" = "$codename" ]; then
      echo
      echo -e "\e31mMarks Has Already Been Recorded For This Course. Please
Try Another.\e[0m"
      echo
     else
      codename=$savedcourse
      newcourse=ok
     fi

    done
```

```
    proceed=i
    until [ "$proceed" = "cont" ]
    do
     echo -en "\e[2mMarks Obtained (0-100)\t\t: "; read marks
     echo -e "\e[0m"

    if ! [[ $marks =~ ^[1]?[0-9]?[0-9]$ ]]; then
     echo
     echo -e "\e[31mINVALID INPUT. PLEASE ENTER POSITIVE INTEGER ONLY.
(0-100)\e[0m"
     echo
    else
     . ./marksscript
    fi
    done

    let qualitypoint=$GPA*$credithour
    let total=$qualitypoint+$total
    totalqp=$(echo "scale=4;$total/10000" |bc)
    point=$(echo "scale=4;$qualitypoint/10000" |bc)

    echo " " >> $id.txt
    echo -e "\t$codename\t\t\t$marks\t\t\t\t$grade\t\t\t$Remarks\t\t$point\t
" >> $studentid.txt

    cont=i
    until [ "$cont" = "ok" ]
    do
    echo
    echo -en "\e[7mDo You Wish To Continue Adding Marks To This Student's
File? YES[Y] NO[N] \e[0m: "; read wish

    case "$wish" in
    [Yy])
          cont=ok
          echo " "
          ;;

    [Nn])
          echo " "
          cont=ok
          next=ok
          echo -e "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t=========" >> $studentid.txt
          echo -e "\t\t\t\t\t\t\t\t\t\t\t\t\tTotal Quality Point: \t$totalqp"
>> $studentid.txt
          echo " "
          echo " "
          echo -en "\e[0m\e[5m\t\t\tSaving The Results File...\e[0m"
          sleep 3
          echo -en "\r\e[0KRECORD SAVED.\n"
          sleep 1
          echo
          echo "RETURNING TO STUDENT VALIDATION FORM..."
          echo
```

```
        sleep 3
        clear
        ./gradescript
        exit
        ;;

    *)
        echo " "
        echo -en "\n\e[31mINVALID OPTION.PLEASE TRY AGAIN.\e[0m"
        echo " "
        ;;
    esac
    done
    done
#The programme will first read the student ID from the user and make sure
the student’s information is in the correct format and have been saved in
the text file before proceeding to add the results.
#Then the programme will check if the student’s id results script has
existed or not. If yes, then will prompt user input whether to overwrite it,
else may leave the add marks form and remain the old history.
#If the student’s id result scripts are new or the student allowed to
overwrite the old results script, academic year, semester, and the course
will be prompted in the correct format.
#The programme will ask the student if they want to enter more course marks
or not. If yes, then the programme will check the second course entered have
duplicated with the first course or not, no duplicated course marks can be
entered into the same student’s results.Else, the programme will sum up the
total quality point of the results script and save it under the student’s ID
text file.
#Students can check if their results exist or not through search results
script from the main menu. Only the correct format of student’s ID and saved
student’s ID results slip will display the output.
```

BACS2093 Operating Systems Assignment Rubrics (session 202101)

1. Student Name : _____
2. Student Name : _____ Programme / Tutorial Group:_____

Task Number	Total marks	Excellent	Good	Average	Poor	Remark
Task 1	**5**	Well-structured program code, comprehensive validations, perfectly correct logic with no bugs, well presentable screen design. **(4-5)**	Good program code structure, most validations provided, correct logic with only minor bugs, good screen design. **(3)**	Reasonable program code structure, some validations provided, correct logic with only minor bugs, reasonable screen design. **(2)**	Poor program code structure, minimal validations provided, some major incorrect logic or major bugs, poor screen design. **(0-1)**	
Task 2	**10**	Well-structured program code, comprehensive validations, perfectly correct logic with no bugs, well presentable screen design. **(9-10)**	Good program code structure, most validations provided, correct logic with only minor bugs, good screen design. **(6-8)**	Reasonable program code structure, some validations provided, correct logic with only minor bugs, reasonable screen design. **(3-5)**	Poor program code structure, minimal validations provided, some major incorrect logic or major bugs, poor screen design. **(0-2)**	
Task 3	**10**	Well-structured program code, comprehensive validations, perfectly correct logic with no bugs, well presentable screen design. **(9-10)**	Good program code structure, most validations provided, correct logic with only minor bugs, good screen design. **(6-8)**	Reasonable program code structure, some validations provided, correct logic with only minor bugs, reasonable screen design. **(3-5)**	Poor program code structure, minimal validations provided, some major incorrect logic or major bugs, poor screen design. **(0-2)**	
Task 4	**15**	Well-structured program code, comprehensive validations, perfectly correct logic with no bugs, well presentable screen design **(12-15)**	Good program code structure, most validations provided, correct logic with only minor bugs, good screen design. **(8-11)**	Reasonable program code structure, some validations provided, correct logic with only minor bugs, reasonable screen design. **(4-7)**	Poor program code structure, minimal validations provided, some major incorrect logic or major bugs, poor screen design. **(0-3)**	
Understanding on program design	**10**	Excellent preparation and delivery of work. A	Adequate preparation and delivery of work. A	Lack of preparation of work and work delivered in average	No preparation of work and work delivered in	

		working system proof of concept that fulfils all the requirements is delivered. **(9-10)**	working system proof of concept that fulfils most of the requirements. **(6-8)**	to below average standard. **(3-5)**	extremely low standard. **(0-2)**	
					Total Marks:	