TUNKU ABDUL RAHMAN UNIVERSITY COLLEGE
FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY

RST3 (Group 2)

Practical Assignment

BACS2093 Operating Systems (AY 202205)

Student Full Name (Block Capital)	Student ID	Contribution (%)	Signature	Marks / 25
1. WONG ZI XIU	21WMR05341	50%		
2. TEH JIN YANG	21WMR05336	50%		

Submission Date: 21 September 2022

Lecturer/Tutor's Name: Ms. CHIN CHAI LIM

BACS2093 Operating Systems Assignment Rubrics (session 202205)

Student Name 1: _Wong Zi Xiu_____

Student Name 2: _Teh Jin Yang_____ Programme / Tutorial Group:_RST3G2_

Course Learning	CLO3: Write shell scripting in UNIX/LINUX commands to create complex programs. (P2, PLO3)

Task Number	**Total marks**	**Excellent**	**Good**	**Average**	**Poor**	**Remark**
Task 1 & Task 2 (Team)	**7**	Well-structured program code, comprehensive validations, perfectly correct logic with no bugs, well presentable screen design. **(6-7)**	Good program code structure, most validations provided, correct logic with only minor bugs, good screen design. **(5)**	Reasonable program code structure, some validations provided, correct logic with only minor bugs, reasonable screen design. **(3-4)**	Poor program code structure, minimal validations provided, some major incorrect logic or major bugs, poor screen design. **(0-2)**	
Task 3 (Team)	**3**	Well-structured program code, comprehensive validations, perfectly correct logic with no bugs, well presentable screen design. **(3)**	Good program code structure, most validations provided, correct logic with only minor bugs, good screen design. **(2)**	Reasonable program code structure, some validations provided, correct logic with only minor bugs, reasonable screen design. **(1)**	Poor program code structure, minimal validations provided, some major incorrect logic or major bugs, poor screen design. **(0)**	

Task 4 (Team)	**10**	Well-structured program code, comprehensive validations, perfectly correct logic with no bugs, well presentable screen design. **(8-10)**	Good program code structure, most validations provided, correct logic with only minor bugs, good screen design. **(6-7)**	Reasonable program code structure, some validations provided, correct logic with only minor bugs, reasonable screen design. **(4-5)**	Poor program code structure, minimal validations provided, some major incorrect logic or major bugs, poor screen design. **(0-3)**	
Understanding on program design (Individual)	**5**	Excellent preparation and delivery of work. A working system proof of concept that fulfils all the requirements is delivered. **(4-5)**	Adequate preparation and delivery of work. A working system proof of concept that fulfils most of the requirements. **(3)**	Lack of preparation of work and work delivered in average to below average standard. **(2)**	No preparation of work and work delivered in extremely low standard. **(0-1)**	Student 1: Student 2:
					Total Marks	Student 1 | Student2

Task 1

1.1 Screen Output(s)

```
================================
   University Management Menu
================================
   1 -> Add New Student
   2 -> Search Student Details
   3 -> Add New Course
   4 -> Search Course Details
   5 -> Grade Student

   Q -> Quit (Exit Program)
================================

Please select a choice (1~5):
```

Figure 1.1

After launching the program, the Menu will be displayed just like the figure above.

```
================================
   University Management Menu
================================
   1 -> Add New Student
   2 -> Search Student Details
   3 -> Add New Course
   4 -> Search Course Details
   5 -> Grade Student

   Q -> Quit (Exit Program)
================================

Please select a choice (1~5): 6
Invalid Input! The Input should between (1~5).
Please Try Again...

Please select a choice (1~5):
```

Figure 1.2

If the input that entered by the user is not within the option range, an error message will be displayed and require the user to input again.

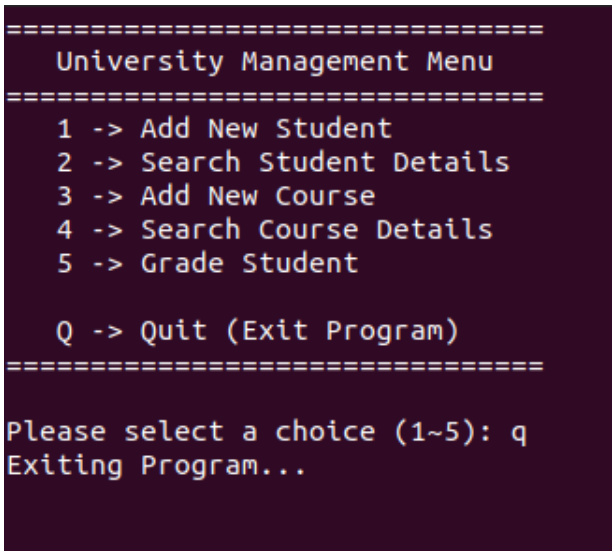

Figure 1.3

If the user entered Q or q, the system will display a message telling the user it is exiting the program.

1.2 Sample Codes (uniMenu)

```
#!/bin/bash
clear
loop=y
      echo "================================"
      echo "   University Management Menu   "
      echo "================================"
      echo "   1 -> Add New Student "
      echo "   2 -> Search Student Details "
      echo "   3 -> Add New Course "
      echo "   4 -> Search Course Details "
      echo "   5 -> Grade Student "
      echo
      echo "   Q -> Quit (Exit Program) "
      echo -e "================================\n"
while [ "$loop" = y ]
do
      echo -n "Please select a choice (1~5): "; read choice

      case "$choice" in
      1) loop=n; sleep 1; clear; ./addNewStud;;
      2) loop=n; sleep 1; clear; ./srchStud;;
      3) loop=n; sleep 1; clear; ./addNewCourse;;
      4) loop=n; sleep 1; clear; ./srchCourse;;
      5) loop=n; sleep 1; clear; ./gradeStud;;
      [qQ]) echo -e "Exiting Program...\n"; sleep 1; clear; exit;;
      *) echo -e "Invalid Input! The Input should between (1~5).\nPlease Try
Again...\n"; loop=y; sleep 1;;
      esac
done
```

In the script uniMenu, the Menu will be displayed and the system will ask for the input. Switch case will be used to check the input. If the user entered the input that is not an option the error message will be displayed and will loop again to ask the user to input again until the user input the option that is available. If user press 1, the system will proceed to adding a new student, if the user press 2, the system will proceed to searching the student details, if the user press 3, the system will proceed add new course, if the user press 4,the system will proceed to search the course details, if they press 5, the system will proceed to the grade student. If they press q or Q the process will delay 1 seconds, clear and exit the programme.

Task 2

2.1 Screen Output(s)

```
============================================================
                  Programme Selection Menu
============================================================
 A -> RIT (Bachelor of Information Techonology)
 B -> RSD (Bachelor of Software Development)
 C -> RST (Bachelor of Interactive Software Techonology)
 D -> REI (Bachelor of Enterprise Information System)
 E -> RSF (Bachelor of Software Engineering)
 F -> RDS (Bachelor of Data Science)
 G -> RIS (Bachelor of Information Security)

 Q -> Quit (Return to University Management Menu)
============================================================

Please select a choice (A~G):
```

Figure 2.1

After the user presses 1 in the main menu, the user will proceed to the programme selection menu to choose the programme of the new student.

```
============================================================
                  Programme Selection Menu
============================================================
 A -> RIT (Bachelor of Information Techonology)
 B -> RSD (Bachelor of Software Development)
 C -> RST (Bachelor of Interactive Software Techonology)
 D -> REI (Bachelor of Enterprise Information System)
 E -> RSF (Bachelor of Software Engineering)
 F -> RDS (Bachelor of Data Science)
 G -> RIS (Bachelor of Information Security)

 Q -> Quit (Return to University Management Menu)
============================================================

Please select a choice (A~G): J
Invalid Input! The Input should between (A~G).
Please Try Again...

Please select a choice (A~G):
```

Figure 2.2

When choosing the programme the input should be between A to G, if the input is not within the option range, the error message will be displayed.

```
============================================================
                    Add New Student Form
============================================================
Student ID (21ABC01234)                  : 21WM05341
Invalid Student ID! Please Try Again...

Student ID (21ABC01234)                  : 21wmr05341
Invalid Student ID! Please Try Again...

Student ID (21ABC01234)                  : 21WMR0534
Invalid Student ID! Please Try Again...

Student ID (21ABC01234)                  : 21WMR05341
Student ID already Exist! Please Try Again...

Student ID (21ABC01234)                  :
```

Figure 2.3

After the user selected the programmes, it will proceed to add new student form. If the id entered was already registered, the system will tell that the id already exists, if the id entered was not in the correct format then it will display invalid student id.

```
============================================================
                    Add New Student Form
============================================================
Student ID (21ABC01234)                  : 21WMR05336
Student Name (same as NRIC)              : Teh Jin Yang
NRIC/Passport Number                     : 012345-67-8901
Birth Date (DD-MM-YYYY)                  : 12-12-2001
Contact Number (012-3456789)             : 012-3456789
Email Address (xxx@student.tarc.edu.my) : tjy@student.tarc.edu.my

Save New Student Details? (Y)es or (N)o : y
Saving Student Details...

Add Another Student? (Y)es or (Q)uit    :
```

Figure 2.4

After the user enters all the information required, the system will ask whether to save the new student details, if yes, the details will be stored in the student.txt (figure 2.5) else nothing will happen. Then the system will ask whether to add another student, if yes, the system will loop again back to (figure 2.1) else return to the main menu.

student.txt

```
Programme:StudentId:StudentName:NRIC/PassportNumber:BirthDate:ContactNumber:EmailAddress
RST:21WMR05341:Wong Zi Xiu:011122-14-0591:22-11-2001:017-6954618:zxwong@student.tarc.edu.my
RST:21WMR05340:Wong Sai Siang:012345-67-8901:12-34-2069:017-6969696:sswong@student.tarc.edu.my
RST:21WMR05339:Wong Rong Kai:098765-12-3456:12-01-2001:012-3456789:rkwong@student.tarc.edu.my
RST:21WMR05336:Teh Jin Yang:012345-67-8901:12-12-2001:012-3456789:tjy@student.tarc.edu.my
```

Figure 2.5

```
=============================================================
                      Search Student Form
=============================================================
Enter Student ID (21ABC01234)           : 21WMR05333
Student ID Not Found! Please Try Again...

Enter Student ID (21ABC01234)           : 21WMR056666
Invalid Student ID! Please Try Again...

Enter Student ID (21ABC01234)           :
```

Figure 2.6

After the user presses 2 in the menu, the user will proceed to the search student form to search the registered student. The system will detect whether input from the user, if the id not yet registered the system will tell that id not found or will display an error message if the id was not in the correct format.

```
Enter Student ID (21ABC01234)           : 21WMR05341


Student Name (auto display)             : Wong Zi Xiu
Contact Number (auto display)           : 017-6954618
Email Address (auto display)            : zxwong@student.tarc.edu.my

=============================================================

Search Another Student? (Y)es or (Q)uit :
```

Figure 2.7

If the registered id entered, the system will display the details of the student such as student name, contact number and email. Then, the system will ask whether to search for another student, if yes, the system will loop again and return to the main menu.

2.2 Sample Codes (addNewStud)

```
#!/bin/bash
again=y
while [ "$again" = y ]
do
            echo "==========================================================="
            echo "                   Programme Selection Menu                "
            echo "==========================================================="
            echo " A -> RIT (Bachelor of Information Technology) "
            echo " B -> RSD (Bachelor of Software Development) "
            echo " C -> RST (Bachelor of Interactive Software Technology)"
            echo " D -> REI (Bachelor of Enterprise Information System) "
            echo " E -> RSF (Bachelor of Software Engineering) "
            echo " F -> RDS (Bachelor of Data Science) "
            echo " G -> RIS (Bachelor of Information Security) "
            echo
            echo " Q -> Quit (Return to University Management Menu) "
            echo -e "============================================================\n"
      progLoop=i
      until [ "$progLoop" = "ok" ]
      do
            echo -n "Please select a choice (A~G): "; read choice

            case "$choice" in
            [aA]) programme="RIT"; progLoop=ok;;
            [bB]) programme="RSD"; progLoop=ok;;
            [cC]) programme="RST"; progLoop=ok;;
            [dD]) programme="REI"; progLoop=ok;;
            [eE]) programme="RSF"; progLoop=ok;;
            [fF]) programme="RDS"; progLoop=ok;;
            [gG]) programme="RIS"; progLoop=ok;;
            [qQ]) echo -e "Returning to University Management Menu...\n";
progLoop=ok; sleep 1; clear; ./uniMenu;;
            *) echo -e "Invalid Input! The Input should between (A~G).\nPlease Try
Again...\n"; progLoop=i; sleep 1;;
            esac
      done

      if [ "$progLoop" = "ok" ]
      then
            echo -e "\n==========================================================="
            echo "                    Add New Student Form                   "
            echo "==========================================================="
            studentId=i
            until [ "$studentId" = "ok" ]
            do
                  echo -n "Student ID (21ABC01234)           : "; read studId

                  if [ ! -f "student.txt" ]
                  then
                        touch student.txt
                  fi

                  studId=$(echo $studId | tr '[a-z]' '[A-Z]')
```

```
                    if [[ $studId =~ ^[0-9]{2}[A-Z]{3}[0-9]{5}$ ]]
                    then
                          compare=$(grep -o "$studId" student.txt)
                          if [ "$studId" = "$compare" ]
                          then
                                echo -e "Student ID already Exist! Please Try
Again...\n"
                          else
                                studentId=ok
                          fi
                    else
                          echo -e "Invalid Student ID! Please Try Again...\n"
                    fi
              done
                    echo -n "Student Name (same as NRIC)              : "; read
studName
                    echo -n "NRIC/Passport Number              : "; read studIc
              birthDate=i
              until [ "$birthDate" = "ok" ]
              do
                    echo -n "Birth Date (DD-MM-YYYY)           : "; read studBd

                    if [[ $studBd =~ ^[0-9]{2}-[0-9]{2}-[0-9]{4}$ ]]
                    then
                          birthDate=ok
                    else
                          echo -e "Invalid Birth Date! Please Try Again...\n"
                    fi
              done
              number=i
              until [ "$number" = "ok" ]
              do
                    echo -n "Contact Number (012-3456789)            : "; read
studNum

                    if [[ $studNum =~ ^[0-9]{3}-[0-9]{7}$ ]]
                    then
                          number=ok
                    else
                          echo -e "Invalid Contact Number! Please Try Again...\n"
                    fi
              done
              email=i
              until [ "$email" = "ok" ]
              do
                    echo -n "Email Address (xxx@student.tarc.edu.my) : "; read
studEmail

                    if [[ $studNum =~ ^[a-z]+@student.tarc.edu.my$ ]]
                    then
                          email=ok
                    else
                          echo -e "Invalid Student Email! Please Try Again...\n"
                    fi
```

```
            done
            saving=i
            until [ "$saving" = "ok" ]
            do
                  echo -en "\nSave New Student Details? (Y)es or (N)o     : ";
read save
                  case "$save" in
                  [yY]) echo "Saving Student Details..."; echo
"$programme:$studId:$studName:$studIc:$studBd:$studNum:$studEmail" >> student.txt;
saving=ok;;
                  [nN]) saving=ok;;
                  *) echo -e "Invalid Input! The Input should be (Y/N).\nPlease
Try Again..."; saving=i; sleep 1;;
                  esac
            done
            adding=i
            until [ "$adding" = "ok" ]
            do
                  echo -en "\nAdd Another Student? (Y)es or (Q)uit : "; read add
                  case "$add" in
                  [yY]) adding=ok; sleep 1; clear; again=y;;
                  [qQ]) echo -e "Returning to University Management Menu...\n";
again=n; adding=ok; sleep 1; clear; ./uniMenu;;
                  *) echo -e "Invalid Input! The Input should be (Y/N).\nPlease
Try Again..."; adding=i; sleep 1;;
                  esac
            done
      fi
done
```

At the choose program part, the switch case will be used to check whether the input alphabet was between A to G or not and both lower and upper case will be accepted. After that, the if statement will check if the progLoop is equal to “ok” or not. If yes, the system will proceed and display the new student form, at the same time it will ask for the input of student id, after the system gets the input it will convert the letter from lowercase to uppercase if the input was lowercase. Then only check whether the student id was in the correct format and whether it has been registered or not. If the student id was new then the system will continue to ask the user to input other details such as student name, student ic, student birth date, student contact number, and student email. The validation is only available for student birth date, student contact number, and student email. After all the input was correct and no error then the system will ask the user whether to save the details or not, if yes, all the details will be appended to the student.txt. Then, the system will ask whether to add another student, if yes, the system will loop again else it will return to Menu.

2.3 Sample Codes (srchStud)

```
#!/bin/bash
```

```
srchLoop=y
while [ "$srchLoop" = y ]
do
      echo "============================================================"
      echo "                     Search Student Form                    "
      echo "============================================================"
      studentId=i
      until [ "$studentId" = "ok" ]
      do
            echo -n "Enter Student ID (21ABC01234)           : "; read studId

            studId=$(echo $studId | tr '[a-z]' '[A-Z]')

            if [[ $studId =~ ^[0-9]{2}[A-Z]{3}[0-9]{5}$ ]]
            then
                  compare=$(grep -o "$studId" student.txt)
                  if [ "$studId" = "$compare" ]
                  then
                        studentId=ok
                  else
                        echo -e "Student ID Not Found! Please Try Again...\n"
                  fi
            else
                  echo -e "Invalid Student ID! Please Try Again...\n"
            fi
      done
            echo -e "\n"
            echo -n "Student Name (auto display)             : "; awk -F ':'
'$2=="'"$studId"'" {print $3}' student.txt
            echo -n "Contact Number (auto display)           : "; awk -F ':'
'$2=="'"$studId"'" {print $6}' student.txt
            echo -n "Email Address (auto display)            : "; awk -F ':'
'$2=="'"$studId"'" {print $7}' student.txt
            echo -e
"\n============================================================"

      again=i
      until [ "$again" = "ok" ]
      do
            echo -en "\nSearch Another Student? (Y)es or (Q)uit    : "; read
search
            case "$search" in
            [yY]) srchLoop=y; studentId=i; again=ok; sleep 1; clear ;;
            [qQ]) echo -e "Returning to University Management Menu...\n";
again=ok; schLoop=n; sleep 1; clear; ./uniMenu;;
            *) echo -e "Invalid Input! The Input should be (Y/N).\nPlease Try
Again..."; again=i; sleep 1;;
            esac
      done
done
```

In the searching student part, the system will accept both lowercase and uppercase while reading the id input. After reading the input, the id will compare to all the student id that grep from the student.txt, if the id entered does not match any of it, the system will display student id not found. If the id was

entered in incorrect format the error message of invalid student id will be displayed and keep loop until the user entered id that is in correct format. If the id entered was correct and found within the student.txt and the details of the student will be displayed and after that the system will ask whether to search for another student, if yes the system will loop again else return to the main menu.

Task 3

3.1 Screen Output (s)

```
============================================================
                    Add New Course Form
============================================================
Course Code (ABCD1234)          : BACS1234
Course already Exist! Please Try Again...

Course Code (ABCD1234)          : BACS12456
Invalid Course Code! Please Try Again...

Course Code (ABCD1234)          :
```

Figure 3.1

After the user presses 3 in the main menu, the user will proceed to add new course to let the user add new course. If the course code that entered was already registered, the system will tell that the course code already exists, if the id entered was not in the correct format then it will display invalid course code.

```
Course Code (ABCD1234)          : BACS4321
Course Name                     : Short Story
Credit Hours (1~12)             : 3
Remarks                         : This is Short Story

Save New Course Details? (Y)es or (N)o  : y
Saving Course Details...

Add Another Course? (Y)es or (Q)uit     :
```

Figure 3.2

After the user enters all the information required, the system will ask whether to save the new course details, if yes, the course details will be stored in the course .txt (figure 3.3) else nothing will happen. Then the system will ask whether to add another course, if yes, the system will loop again else return to the main menu.

```
course.txt
CourseCode:CourseName:CreditHour(s):Remarks
BACS1234:Information Technology:4:This course will offer Bachelor Degree Year 1
BACS2053:Object-Oriented Programming:3:This course will offer Bachelor Degree Year 2
BACS2093:Operating System:3:This course will offer every semester
BACS2073:Mobile Application Development:4:This is MAD
BACS3033:Social and Professional Issues:3:This is SPI
BACS4321:Short Story:3:This is Short Story
```

Figure 3.3

```
============================================================
                    Search Course Form
============================================================
Enter Course Code (ABCD1234)            : BACS1235
Course Not Found! Please Try Again...

Enter Course Code (ABCD1234)            : BACS12345
Invalid Course Code! Please Try Again...

Enter Course Code (ABCD1234)            : BACS2093
```

Figure 3.4

After the user presses 4 in the menu, the user will proceed to the search course form to search the registered course. The system will detect whether input from the user, if the course code is not yet registered the system will tell that id not found or will display an error message if the course code was not in the correct format.

```
Enter Course Code (ABCD1234)            : BACS2093


Course Name (auto display)              : Operating System
Credit Hour(s) (auto display)           : 3
Remarks (auto display)                  : This course will offer every semester

============================================================

Search Another Course? (Y)es or (Q)uit  :
```

Figure 3.5

If the registered course code entered, the system will display the details of the course such as course name, credit hours and remark. Then, the system will ask whether to search for another course , if yes, the system will loop again and return to the main menu.

3.2 Sample Codes (addNewCourse)

```
#!/bin/bash
again=y
while [ "$again" = y ]
do
      echo "============================================================"
      echo "                    Add New Course Form                   "
      echo "============================================================"
      code=i
      until [ "$code" = "ok" ]
      do
            echo -n "Course Code (ABCD1234)            : "; read courseCode

            if [ ! -f "course.txt" ]
            then
                  touch course.txt
            fi

            courseCode=$(echo $courseCode | tr '[a-z]' '[A-Z]')

            if [[ $courseCode =~ ^[A-Z]{4}[0-9]{4}$ ]]
            then
                  compare=$(grep -o "$courseCode" course.txt)
                  if [ "$courseCode" = "$compare" ]
                  then
                        echo -e "Course already Exist! Please Try Again...\n"
                  else
                        code=ok
                  fi
            else
                  echo -e "Invalid Course Code! Please Try Again...\n"
            fi
      done
            echo -n "Course Name                 : "; read courseName
      hour=i
      until [ "$hour" = "ok" ]
      do
            echo -n "Credit Hours (1~12)         : "; read creditHour

            if [ "$creditHour" -lt 1 -o "$creditHour" -gt 12 ]
            then
                  echo -e "Invalid Credit Hour(s)! The Input should between
(1~12).\nPlease Try Again...\n"
            else
                  hour=ok
            fi
      done
            echo -n "Remarks                     : "; read remarks
      saving=i
      until [ "$saving" = "ok" ]
      do
            echo -en "\nSave New Course Details? (Y)es or (N)o     : "; read save
            case "$save" in
            [yY]) echo "Saving Course Details..."; echo
```

```
"$courseCode:$courseName:$creditHour:$remarks" >> course.txt; saving=ok;;
            [nN]) saving=ok;;
            *) echo -e "Invalid Input! The Input should be (Y/N).\nPlease Try
Again..."; saving=i; sleep 1;;
            esac
      done
      adding=i
      until [ "$adding" = "ok" ]
      do
            echo -en "\nAdd Another Course? (Y)es or (Q)uit  : "; read add
            case "$add" in
            [yY]) adding=ok; sleep 1; clear; again=y;;
            [qQ]) echo -e "Returning to University Management Menu...\n";
adding=ok; again=i; sleep 1; clear; ./uniMenu;;
            *) echo -e "Invalid Input! The Input should be (Y/N).\nPlease Try
Again..."; adding=i; sleep 1;;
            esac
      done
done
```

In the add new course form, the system requires the user to input the course code, the system accepts both lower and upper case. After the system gets the input from the user it will convert the letter from lowercase to uppercase if the input was lowercase. Then only check whether the course code was in the correct format and whether it has been registered or not. If the course code was new then the system will continue to ask the user to input other details such as course name, credit hours, remark. The credit hours have the validation whether the input number was within 1 to 12 or not. After all the input was correct and no error then the system will ask the user whether to save the course details or not, if yes, all the details will be appended to the course.txt. Then, the system will ask whether to add another course, if yes, the system will loop again else it will return to Menu.

3.3 Sample Codes (srchCourse)

```
#!/bin/bash
srchLoop=y
while [ "$srchLoop" = y ]
do
      echo "============================================================"
      echo "                     Search Course Form                     "
      echo "============================================================"
      code=i
      until [ "$code" = "ok" ]
      do
            echo -n "Enter Course Code (ABCD1234)            : "; read courseCode

            courseCode=$(echo $courseCode | tr '[a-z]' '[A-Z]')

            if [[ $courseCode =~ ^[A-Z]{4}[0-9]{4}$ ]]
            then
                  compare=$(grep -o "$courseCode" course.txt)
                  if [ "$courseCode" = "$compare" ]
                  then
                        code=ok
                  else
                        echo -e "Course Not Found! Please Try Again...\n"
                  fi
            else
                  echo -e "Invalid Course Code! Please Try Again...\n"
            fi
      done
            echo -e "\n"
            echo -n "Course Name (auto display)        : "; awk -F ':'
'$1=="'"$courseCode"'" {print $2}' course.txt
            echo -n "Credit Hour(s) (auto display)           : "; awk -F ':'
'$1=="'"$courseCode"'" {print $3}' course.txt
            echo -n "Remarks (auto display)                  : "; awk -F ':'
'$1=="'"$courseCode"'" {print $4}' course.txt
            echo -e "\n============================================================"

      again=i
      until [ "$again" = "ok" ]
      do
            echo -en "\nSearch Another Course? (Y)es or (Q)uit     : "; read
search
            case "$search" in
            [yY]) srchLoop=y; code=i; again=ok; sleep 1; clear;;
            [qQ]) echo -e "Returning to University Management Menu...\n";
again=ok; srchLoop=n; sleep 1; clear; ./uniMenu;;
            *) echo -e "Invalid Input! The Input should be (Y/N).\nPlease Try
Again..."; again=i; sleep 1;;
            esac
      done
done
```

In the searching course part, the system will accept both lowercase and uppercase while reading the id input. After reading the input, the id will compare to all the course code that grep from the

course.txt, if the course code entered does not match any of it, the system will display course code not found. If the course code was entered in incorrect format the error message of invalid course code will be displayed and keep loop until the user entered id that is in correct format. If the id entered was correct and found within the course.txt and the details of the course will be displayed and after that the system will ask whether to search for another course, if yes the system will loop again else return to the main menu.

Task 4

4.1 Screen Output(s)

```
=============================================================
                    Student Validation Form
=============================================================
Enter Student's ID Number                : 21WMR05320
Student ID Not Found! Please Try Again...

Enter Student's ID Number                : 21WMR054678
Invalid Student ID! Please Try Again...

Enter Student's ID Number                : 
```

Figure 4.1

After the user presses 5 in the main menu, the user will proceed to grade the student to let them grade the student. The system will detect whether input from the user, if the id is not yet registered the system will tell that id not found or will display an error message if the id was not in the correct format.

```
Enter Student's ID Number                : 21WMR05341

Student Name (auto display)              : Wong Zi Xiu
Programme (auto display)                 : RST
Academic Year (YYYY)                     : 2021
Semester (1/2/3)                         : 2
=========================================================

Press (c) to continue enter student's mark(s) or (q) to quit from the prompt    :
```

Figure 4.2

If the registered id entered, the system will display the details of the student such as student name, and programme. After that, it will require the user to enter the academic year and the semester, when the user enters all the information required correctly, the system will ask the user whether to continue to enter student's marks or quit the main menu.

```
============================================================
             Student Examination Mark(s) Form
============================================================
Enter Course Code                        : BACS2093
Course Name (auto display)               : Operating System

Enter the mark(s) obtained               : 90
============================================================

Press (c) to continue enter student's mark(s) or (g) to generate exam result    : c

Enter Course Code                        : bacs2093
The Course has been added! Please Try Another Course...

Enter Course Code                        : bacs1235
Course Not Found! Please Try Again...

Enter Course Code                        :
```

Figure 4.3

If the user pressed c, the system will proceed to the student examination mark form which require the user to enter the course code, the course code entered will be check whether it exist or not, if yes, it will let the user to input the mark of the student and store into tempResult.txt (figure 4.4). If the user accidentally enter same course code the error message will be displayed as well as entered the wrong course code. Then, it will ask the user whether to continue or generate exam result.

```
tempResult.txt
1 BACS2093:90:A:Excellent:12.000
```

Figure 4.4

```
============================================================
             Student Examination Result(s) Form
============================================================
 Student ID                              : 21WMR05341
 Student Name                            : Wong Zi Xiu
 Academic Year                           : 2021
 Semester                                : 2
============================================================

==========================================================================================
 Course Code:   Mark(s) Obtained:       Grade Obtained:         Remark(s):      Quality Point:
==========================================================================================
 BACS2093       90                      A                       Excellent       12
 BACS2073       70                      B+                      Good            14
 BACS1234       55                      C+                      Pass            10

                                                   Total Quality Point(s):      36
==========================================================================================

Press (q) to return to University Management Menu       :
```

Figure 4.5

If the user pressed g which is to generate exam results, the figure 4.5 above will be the output of the student which includes all information that entered previously and the total quality point will be calculated. Lastly, it will ask the user to press q to return to the main menu.

21WMR05341Result.txt

```
============================================================
             Student Examination Result(s) Form
============================================================
 Student ID                           : 21WMR05341
 Student Name                         : Wong Zi Xiu
 Academic Year                        : 2021
 Semester                             : 2
============================================================

==========================================================================================
 Course Code:   Mark(s) Obtained:      Grade Obtained:                Remark(s):     Quality Point:
==========================================================================================
 BACS2093       90                      A                        Excellent       12
 BACS2073       70                      B+                       Good            14
 BACS1234       55                      C+                       Pass            10

                                                [1mTotal Quality Point(s):        36[0m
==========================================================================================
```

Figure 4.6

Figure 4.6 is the student's examination result but stored inside an txt file.

4.2 Sample Codes (marks)

```
#!/bin/bash
# $1=coursecode, $2=mark, $3=credithour

if [ $2 -ge 80 -a $2 -le 100 ]
then
      grade=A
      gradePoint=4.000
      remarks=Excellent
elif [ $2 -ge 75 -a $2 -le 79 ]
then
      grade=A-
      gradePoint=3.750
      remarks=Excellent
elif [ $2 -ge 70 -a $2 -le 74 ]
then
      grade=B+
      gradePoint=3.500
      remarks=Good
elif [ $2 -ge 65 -a $2 -le 69 ]
then
      grade=B
      gradePoint=3.000
      remarks=Good
elif [ $2 -ge 60 -a $2 -le 64 ]
then
      grade=B-
      gradePoint=2.750
      remarks=Pass
elif [ $2 -ge 55 -a $2 -le 59 ]
then
      grade=C+
      gradePoint=2.500
      remarks=Pass
elif [ $2 -ge 50 -a $2 -le 54 ]
then
      grade=C
      gradePoint=2.000
      remarks=Pass
elif [ $2 -ge 0 -a $2 -le 49 ]
then
      grade=F
      gradePoint=0.000
      remarks=Failed
elif [ $2 -gt 100 -o $2 -lt 1 ]
then
      echo "Invalid Mark(s) Range!"
fi

qualityPoint=`echo "var=$gradePoint; var*=$3; var" | bc`

if [ ! -f "tempResult.txt" ]
then
      touch tempResult.txt
```

```
fi

echo "$1:$2:$grade:$remarks:$qualityPoint" >> tempResult.txt
```

In the marks script it will read the courseCode, marks and the creditHour. After that, the marks will go through the if statement and return the grade, gradePoint and the remarks. Then, it will calculate the quality point by using gradePoint multiplied by creditHour, then the data such as courseCode, marks, grade, remarks and the qualityPoint will be stored in the tempResult.txt.

4.3 Sample Codes (gradeStud)

```
#!/bin/bash
again=y
while [ "$again" = y ]
do
            echo "============================================================"
            echo "                  Student Validation Form                  "
            echo "============================================================"
      studentId=i
      until [ "$studentId" = "ok" ]
      do
            echo -n "Enter Student's ID Number         : "; read studId

            studId=$(echo $studId | tr '[a-z]' '[A-Z]')

            if [[ $studId =~ ^[0-9]{2}[A-Z]{3}[0-9]{5}$ ]]
            then
                  compare=$(grep -o "$studId" student.txt)
                  if [ "$studId" = "$compare" ]
                  then
                        studentId=ok
                  else
                        echo -e "Student ID Not Found! Please Try Again...\n"
                  fi
            else
                  echo -e "Invalid Student ID! Please Try Again...\n"
            fi
      done
            studName=`awk -F ':' '$2=="'"$studId"'" {print $3}' student.txt`
            echo
            echo -en "Student Name (auto display)            : $studName \n";
            echo -n "Programme (auto display)          : "; awk -F ':'
'$2=="'"$studId"'" {print $1}' student.txt
      year=i
      until [ "$year" = "ok" ]
      do
            echo -n "Academic Year (YYYY)              : "; read acaYear

            if [[ $acaYear =~ ^[0-9]{4}$ ]]
            then
                  year=ok
            else
```

```
                  echo -e "Invalid Academic Year! Please Try Again...\n"
            fi
      done
      sem=i
      until [ "$sem" = "ok" ]
      do
            echo -n "Semester (1/2/3)                  : "; read semester

            if [ "$semester" -lt 1 -o "$semester" -gt 3 ]
            then
                  echo -e "Invalid Semester! The Input should be (1/2/3).\nPlease
Try Again...\n"
            else
                  sem=ok
            fi
      done

            echo -e
"============================================================\n"
      con=i
      until [ "$con" = "ok" ]
      do
            echo -en "Press (c) to continue enter student's mark(s) or (q) to quit
from the prompt   : "; read corq
            case "$corq" in
            [cC]) con=ok; sleep 1;;
            [qQ]) echo -e "Returning to University Management Menu...\n"; con=ok;
sleep 1; clear; ./uniMenu;;
            *) echo -e "Invalid Input! The Input should be (Y/N).\nPlease Try
Again..."; con=i; sleep 1;;
            esac
      done
            echo
            echo "============================================================"
            echo "             Student Examination Mark(s) Form              "
            echo "============================================================"
      count=0
      again=i
      dupes=n
      until [ "$again" = "ok" ]
      do
            #echo "$count"
            code=i
            until [ "$code" = "ok" ]
            do
                  echo -n "Enter Course Code                 : "; read courseCode

                  courseCode=$(echo $courseCode | tr '[a-z]' '[A-Z]')

                  if [[ $courseCode =~ ^[A-Z]{4}[0-9]{4}$ ]]
                  then
                        if [ "$count" -ge 1 ]
                        then
                              compareTmp=$(grep -o "$courseCode" tempResult.txt)
                              if [ "$courseCode" = "$compareTmp" ]
```

```
                                then
                                      echo -e "The Course has been added! Please
Try Another Course...\n"
                                      dupes=y
                                else
                                      dupes=n
                                fi
                        fi
                        if [ "$dupes" = n ]
                        then
                                compare=$(grep -o "$courseCode" course.txt)
                                if [ "$courseCode" = "$compare" ]
                                then
                                      code=ok
                                else
                                      echo -e "Course Not Found! Please Try
Again...\n"
                                fi
                        fi
                  else
                        echo -e "Invalid Course Code! Please Try Again...\n"
                  fi
            done
                  echo -n "Course Name (auto display)        : "; awk -F ':'
'$1=="'"$courseCode"'" {print $2}' course.txt
                  echo
            mark=i
            until [ "$mark" = "ok" ]
            do
                  echo -n "Enter the mark(s) obtained        : "; read score

                  if [ "$score" -lt 0 -o "$score" -gt 100 ]
                  then
                        echo -e "Invalid Mark(s) Range! The Mark(s) should between
(0~100) Please Try Again...\n"

                  else
                        mark=ok
                  fi
            done
                  creditHours=`awk -F ':' '$1=="'"$courseCode"'" {print $3}'
course.txt`
                  ./marks $courseCode $score $creditHours
                  echo -e
"============================================================\n"
            continue=i
            until [ "$continue" = "ok" ]
            do
                  echo -en "Press (c) to continue enter student's mark(s) or (g)
to generate exam result  : "; read corg
                  case "$corg" in
                  [cC]) count=$((count+1)); echo;  continue=ok; again=i; sleep 1;;
                  [gG]) echo -e "Generating Exam Result...\n"; continue=ok;
again=ok; sleep 1;;
                  *) echo -e "Invalid Input! The Input should be (Y/N).\nPlease
```

```
Try Again..."; continue=i; sleep 1;;
                  esac
            done
      done
            if [[ $corg =~ ^[gG]$ ]]
            then
                  filename="$studId"Result
                  if [ -f "$filename".txt ]
                  then
                        rm "$filename".txt
                        touch "$filename".txt
                  fi

                  clear
                  echo
"============================================================" >> "$filename".txt
                  echo "             Student Examination Result(s) Form
" >> "$filename".txt
                  echo
"============================================================" >> "$filename".txt
                  echo " Student ID                    : $studId " >>
"$filename".txt
                  echo " Student Name                        : $studName " >>
"$filename".txt
                  echo " Academic Year                       : $acaYear " >>
"$filename".txt
                  echo " Semester                      : $semester " >>
"$filename".txt
                  echo
"============================================================" >> "$filename".txt

                  echo -e
"\n=================================================================================
================" >> "$filename".txt
                  echo -e " Course Code:\tMark(s) Obtained:\tGrade
Obtained:\t\tRemark(s):\tQuality Point: " >> "$filename".txt
                  echo
"=================================================================================
==============" >> "$filename".txt
                  result=`awk -F ':' '{printf " %-14s %-23d %-23s %-15s %g\n", $1,
$2, $3, $4, $5}' tempResult.txt`
                  echo "$result" >> "$filename".txt
                  sum=`awk -F ':' '{sum+=$5} END{print sum}' tempResult.txt`
                  echo -e "\n\t\t\t\t\t\t   \033[1mTotal Quality Point(s):
\t$sum\033[0m" >> "$filename".txt
                  rm tempResult.txt
                  echo
"=================================================================================
==============" >> "$filename".txt
            fi

            cat "$filename".txt

            return=i
            until [ "$return" = "ok" ]
```

```
            do
                  echo -en "\nPress (q) to return to University Management Menu :
"; read back
                  case "$back" in
                  [qQ]) echo -e "Returning to University Management Menu...\n";
return=ok; sleep 1; ./uniMenu;;
                  *) echo -e "Invalid Input! The Input should be (Y/N).\nPlease
Try Again..."; return=i; sleep 1;;
                  esac
            done
done
```

In the grade student part, the system asks for the student id and will accept both lowercase and uppercase while reading the id input. After reading the input, the id will compare to all the student id that grep from the student.txt, if the id entered does not match any of it, the system will display student id not found. If the id was entered in incorrect format the error message of invalid student id will be displayed and keep loop until the user entered id that is in correct format. If the id entered was correct and found within the student.txt and the student name and its programme will be displayed. After that, the user is required to enter the academic year and the semester of the user, while the academic year has the validation of must be starting with 20 while the other 2 numbers behind can be any number with the range 0 to 9. The semester also had a validation of whether the input was between 1 to 3.

After that the user can choose whether press c continues to grade the student or press q to return to the main menu. Everytime the user grade the student the counter will increase by 1 and the check will start after the counter greater equal 1 and if the course that has been graded being entered again the system will display error message and all the course that graded the marks will be pass into the marks script and calculate the quality points then only will be stored inside the tempResult.txt. When generate the exam result the information that entered just now will be display include and the result such as marks, grade, remarks, qualityPoint will be get from the tempResult.txt and the generated exam result will be store inside the file that renamed as the studentIDResult.txt After that, the system will ask the user to press q to return to the main menu.