I. Code File and Executable: The original binary file that you are provided with should be converted to a fully running program in C that functions as described in your commented code.

A. Convert the binary file into assembly code.

FinalProject.o

main()

push %rbp

mov %rsp,%rbp

sub $0x10,%rsp

mov %edi,-0x4(%rbp)

mov %rsi,-0x10(%rbp)

mov $0x0,%eax

callq 0x4005ed <ReadUserInfo>

mov $0x0,%eax

callq 0x40062f <CheckUserPermissionAccess>

cmp $0x1,%eax

jne 0x4007be <main+260>

mov $0x400888,%edi

callq 0x4004b0 <puts@plt>

mov $0x0,%eax

callq 0x40064c <DisplayStudentInformation>

mov $0x4008b8,%edi

callq 0x4004b0 <puts@plt>

mov $0x6010e8,%esi

mov $0x400864,%edi

mov $0x0,%eax

callq 0x4004f0 <\_\_isoc99\_scanf@plt>

movzbl 0x2009c9(%rip),%eax # 0x6010e8 <choice>

cmp $0x59,%al

jne 0x4007be <main+260>

mov $0x4008d8,%edi

mov $0x0,%eax

callq 0x4004c0 <printf@plt>

movl $0x0,0x2009ac(%rip) # 0x6010ec <i>

jmp 0x40079f <main+229>

mov 0x2009a4(%rip),%eax # 0x6010ec <i>

movslq %eax,%rdx

mov %rdx,%rax

shl $0x2,%rax

add %rdx,%rax

add %rax,%rax

add $0x601080,%rax

mov %rax,%rsi

mov $0x400864,%edi

mov $0x0,%eax

callq 0x4004c0 <printf@plt>

mov 0x200976(%rip),%eax # 0x6010ec <i>

cltq

add $0x601060,%rax

mov %rax,%rsi

mov $0x400864,%edi

mov $0x0,%eax

callq 0x4004f0 <\_\_isoc99\_scanf@plt>

mov 0x200956(%rip),%eax # 0x6010ec <i>

add $0x1,%eax

mov %eax,0x20094d(%rip) # 0x6010ec <i>

mov 0x200947(%rip),%eax # 0x6010ec <i>

cmp $0x4,%eax

jle 0x400742 <main+136>

mov $0x400908,%edi

callq 0x4004b0 <puts@plt>

mov $0x0,%eax

callq 0x40064c <DisplayStudentInformation>

mov $0x0,%eax

leaveq

retq

DisplayStudentInformation()

push %rbp

mov %rsp,%rbp

movl $0x0,0x200a92(%rip) # 0x6010ec <i>

jmp 0x4006ad <DisplayStudentInformation+97>

mov 0x200a8a(%rip),%eax # 0x6010ec <i>

cltq

movzbl 0x601060(%rax),%eax

movsbl %al,%ecx

mov 0x200a78(%rip),%eax # 0x6010ec <i>

movslq %eax,%rdx

mov %rdx,%rax

shl $0x2,%rax

add %rdx,%rax

add %rax,%

add $0x601080,%rax

mov %ecx,%edx

mov %rax,%rsi

mov $0x40087a,%edi

mov $0x0,%eax

callq 0x4004c0 <printf@plt>

mov 0x200a48(%rip),%eax # 0x6010ec <i>

add $0x1,%eax

mov %eax,0x200a3f(%rip) # 0x6010ec <i>

mov 0x200a39(%rip),%eax # 0x6010ec <i>

cmp $0x4,%eax

jle 0x40065c <DisplayStudentInformation+16>

pop %rbp

retq

CheckUserPermisssionAccess()

push %rbp

mov %rsp,%rbp

mov 0x200aab(%rip),%eax # 0x6010e4 <password>

cmp $0x7b,%eax

jne 0x400645 <CheckUserPermissionAccess+22>

mov $0x1,%eax

jmp 0x40064a <CheckUserPermissionAccess+27>

mov $0x0,%eax

pop %rbp

retq

ReadUserInfo()

push %rbp

mov %rsp,%rbp

mov $0x400858,%edi

callq 0x4004b0 <puts@plt>

mov $0x6010d0,%esi

mov $0x400864,%edi

mov $0x0,%eax

callq 0x4004f0 <\_\_isoc99\_scanf@plt>

mov $0x400867,%edi

callq 0x4004b0 <puts@plt>

mov $0x6010e4,%esi

mov $0x400877,%edi

mov $0x0,%eax

callq 0x4004f0 <\_\_isoc99\_scanf@plt>

pop %rbp

retq

B. Assembly code is properly commented and explained.

|  |  |
| --- | --- |
| **Assembly Code** | **Explanation** |
| main()  push %rbp  mov %rsp,%rbp | Beginning of the function that copies the  stack pointer to the base pointer. executes the main() |
| sub $0x10,%rsp  mov %edi,-0x4(%rbp)  mov %rsi,-0x10(%rbp)  mov $0x0,%eax  callq 0x4005ed <ReadUserInfo> | Creates and Initializes variables  Utilizes temporary storage  Calls the ReadUserInfo function |
| mov $0x0,%eax  callq 0x40062f <CheckUserPermissionAccess>  cmp $0x1,%eax  jne 0x4007be <main+260>  mov $0x400888,%edi  callq 0x4004b0 <puts@plt> | Calls the CheckUserPermission function  Compares user input to value stored  Jumps to main if condition is met(user credentials valid)  Calls the data stored once acess is granted |
| mov $0x0,%eax  callq 0x40064c <DisplayStudentInformation>  mov $0x4008b8,%edi  callq 0x4004b0 <puts@plt> | Calls the DisplayStudentInformation function  Calls the data stored to access student info |
| mov $0x6010e8,%esi  mov $0x400864,%edi  mov $0x0,%eax  callq 0x4004f0 <\_\_isoc99\_scanf@plt> | Moves values stored for student grades to temporary registers  Accepts input from user |
| movzbl 0x2009c9(%rip),%eax # 0x6010e8 <choice>  cmp $0x59,%al  jne 0x4007be <main+260>  mov $0x4008d8,%edi  mov $0x0,%eax  callq 0x4004c0 <printf@plt> | Input from user to choose to edit student info or not  Compares value stored to input to determine what to do next  Jumps to main to edit  Prints the result of user input |
| movl $0x0,0x2009ac(%rip) # 0x6010ec <i>  jmp 0x40079f <main+229>  mov 0x2009a4(%rip),%eax # 0x6010ec <i>  movslq %eax,%rdx | Students grades are stored in temporary registers  Moves stored values and user input |
| mov %rdx,%rax  shl $0x2,%rax  add %rdx,%rax  add %rax,%rax  add $0x601080,%rax  mov %rax,%rsi  mov $0x400864,%edi  mov $0x0,%eax  callq 0x4004c0 <printf@plt> | Values stored and user input are moved to temporary registers  Computation is performed to add new values to existing values  Prints the results of the student's new grade |
| mov 0x200976(%rip),%eax # 0x6010ec <i>  cltq | Loop to prompt user to enter new grades |
| add $0x601060,%rax  mov %rax,%rsi  mov $0x400864,%edi  mov $0x0,%eax  callq 0x4004f0 <\_\_isoc99\_scanf@plt> | Continues the loop for the number of students  Reads input from user to change grades |
| mov 0x200956(%rip),%eax # 0x6010ec <i>  add $0x1,%eax  mov %eax,0x20094d(%rip) # 0x6010ec <i>  mov 0x200947(%rip),%eax # 0x6010ec <i>  cmp $0x4,%eax  jle 0x400742 <main+136>  mov $0x400908,%edi  callq 0x4004b0 <puts@plt> | Continues the loop for the number of students  Compares the number of students to the iterations  Jumps to main if nor more students  Reads input from user to change grades |
| mov $0x0,%eax  callq 0x40064c <DisplayStudentInformation>  mov $0x0,%eax | exits the loop  call the DisplayStudent Information function |
| leaveq  retq | Jumps back to the return address |
| DisplayStudentInformation()  push %rbp  mov %rsp,%rbp | Beginning of the function that copies the  stack pointer to the base pointer |
| movl $0x0,0x200a92(%rip) # 0x6010ec <i>  jmp 0x4006ad <DisplayStudentInformation+97>  mov 0x200a8a(%rip),%eax # 0x6010ec <i> | Jumps to DisplayStudentInformation |
| cltq  movzbl 0x601060(%rax),%eax  movsbl %al,%ecx  mov 0x200a78(%rip),%eax # 0x6010ec <i>  movslq %eax,%rdx  mov %rdx,%rax | Loops through an array of student names and grades |
| shl $0x2,%rax  add %rdx,%rax  add %rax,%  add $0x601080,%rax  mov %ecx,%edx  mov %rax,%rsi  mov $0x40087a,%edi  mov $0x0,%eax  callq 0x4004c0 <printf@plt> | Assigns grades to students in corresponding locations  Registers where student's names and grades are stored  Prints the values of student names and corresponding grades |
| mov 0x200a48(%rip),%eax # 0x6010ec <i>  add $0x1,%eax  mov %eax,0x200a3f(%rip) # 0x6010ec <i>  mov 0x200a39(%rip),%eax # 0x6010ec <i>  cmp $0x4,%eax  jle 0x40065c <DisplayStudentInformation+16> | Counter for loop  Compares values stored  Jumps to DisplayStudentInformation if condition is met |
| pop %rbp  retq | Beginning of the function that copies the  stack pointer to the base pointer |
| CheckUserPermissionAccess()  push %rbp  mov %rsp,%rbp | Beginning of the function that copies the  stack pointer to the base pointer. Executes the CheckUserPermission() |
| mov 0x200aab(%rip),%eax # 0x6010e4 <password>  cmp $0x7b,%eax  jne 0x400645 <CheckUserPermissionAccess+22> | Takes input from user for password  Compares value stored %eax to what was entered  Jumps to verify user's credentials |
| mov $0x1,%eax  jmp 0x40064a <CheckUserPermissionAccess+27>  mov $0x0,%eax | Returns the value stored in %eax to continue to program or exit if user  password is verified or permission denied |
| pop %rbp  retq | Pops the base pointer off the stack and  stores it in %rbp and jumps back to return  address |
| ReadUserInfo()  push %rbp  mov %rsp,%rbp | Beginning of the function that copies the  stack pointer to the base pointer. Executes the ReadUserInfo() |
| mov $0x400858,%edi  callq 0x4004b0 <puts@plt>  mov $0x6010d0,%esi  mov $0x400864,%edi  mov $0x0,%eax  callq 0x4004f0 <\_\_isoc99\_scanf@plt> | Calls the function to display "Enter Name"  Reads input for user name |
| mov $0x400867,%edi  callq 0x4004b0 <puts@plt>  mov $0x6010e4,%esi  mov $0x400877,%edi  mov $0x0,%eax  callq 0x4004f0 <\_\_isoc99\_scanf@plt> | Calls the function to display "Enter password"  Reads input for user password |
| pop %rbp  retq | Beginning of the function that copies the  stack pointer to the base pointer |

C. Disassembled code is completely converted into higher-order programming language (C code).

D. Translated code’s primary functions are properly commented out.

E. Translated code is logically organized and primary functions execute properly.

#include <stdio.h>

#include <stdlib.h>

// initializes the variables

char name[] = "";

char password[] ="";

char \*students[] = {"Jim", "Tom", "Ben", "Alice", "Ruby"};

char grades[] = "ACCDF";

int main(int argc, char\*\* argv) {

ReadUserInfo(); // calls the function to display messages and take user name and password

int access = CheckUserPermissionAcess(); // unction to check credentials

if(access == 1) {

DisplayStudentInformation(); // displays student's names and grades

printf("Adjust the grades for students?"); // prompts the user to make a selection

char userChoice = 'Y';

char userValue = 'X';

scanf(" %c", &uservalue); // reads input from user

if(userChoice == userValue) {

printf("Enter the GPA for students one a t a time \n");

int i;

for(i =0; i <5; i++) {

printf("%s", students[i]);

scanf("%c", &grades[i]); // reads user input to change grades

}

}

printf("You have successfully updated class grades. The grades are now as follows:\n")

DisplayStudentInformation();

}else {

return(0);

}

return(EXIT\_SUCCESS);

void DisplayStudentInformation() {

int i;

for(i =0; i < 5; i ++) {

printf("%s %c \n", students[i], grades[i]);

}

}

int CheckUserPermissionAccess() {

char pass[] = "123";

char name[] = " ";

if(strcmp(password, pass) == 0) { // compares input to password

return 1; // returns 1 to DisplayStudentInformation

} else{

return 0; // exits the program

}

}

void ReadUserInfo() {

printf("Enter Name: \n");

scanf("%s, &name); // reads input for user name

printf("Enter Password: \n");

scanf("%s", &password); // reads input for user password

}