INDIVIDUAL ASSIGNMENT

TECHNOLOGY PARK MALAYSIA

CT018-3-1-ICP

INTRODUCTION TO C PROGRAMMING

APU1F2009CS(DA)

NAME: MARCELL AGUNG WAHYUDI

TP NUMBER: TP058650

HAND OUT DATE: 24 MAY 2021

HAND IN DATE: 28 JUNE 2021

WEIGHTAGE: 50%

Contents

1. Introduction.....3

2. Assumptions.....4

3. Design of the Program.....5

3.1 Design - Pseudocode.....5

3.2 Design – Flowchart.....18

4. Additional Features.....24

5. Screenshot of Input/Output.....25

6. Conclusion.....32

7. References.....33

1. Introduction

Coronavirus, a virus that causes coronavirus disease (COVID-19) is a very lethal virus that has caused global pandemic, this virus infects the respiratory system, the result of getting infected by the coronavirus ranges from light to fatal.

As of today, there are no effective medication to completely cure patients who has been affected by corona. Thus, taking the Covid-19 vaccine will help protect against this virus, but because the virus is all over the world, this makes pharmaceutical companies to compete against time to develop vaccine. The vaccination progress has started in numerous countries, including Malaysia. The type of vaccine planned for people in Malaysia are Pfizer, Sinovac, AstraZeneca, SputnikV, and CanSinoBio.

The pharmaceutical company is in need of a system to manage the vaccine inventory with mandatory functions such as inventory creation, vaccine quantity updation, vaccine quantity search and produce distributed vaccine quantity list.

The System will have a permanent record to store data, one of which is vaccine.txt in order to store the details of a vaccine (name of vaccine, vaccine code, producing country, dosage required, population covered, and vaccine quantity (in millions)), the system will also be able to update the vaccine quantities if the vaccine were to be added or distributed, and for convenience sake, the system will be able to search a vaccine quantity just by searching it’s code which will simplify the employee’s work. And finally the system will also have a record for vaccine distribution which will be stored in dist.txt, this list will be sorted descendingly (highest quantity to lowest quantity).

2. Assumptions

1. It is assumed the system is able to create a new vaccine inventory, the new vaccine record will be appended to vaccine.txt
2. It is assumed the system is able to update vaccine quantities. The system will have a function to select a specific vaccine and specify either received or distributed quantity. The quantity of the specified vaccine will then be added or subtracted correspondingly in the vaccine.txt file. E.g., suppose the initial quantity of vaccine A in vaccine.txt is 3 million. When the company is distributing 1 million vaccines to a hospital, the quantity of vaccine A will be updated to be 2 million instead of 3 million. This distribution will also be recorded into a file called dist.txt.
3. It is assumed the system is able to search a vaccine’s available quantity by using vaccine code, the system is assumed to have a feature where an employee to search for a specific vaccine’s quantity by inputting only the vaccine code.
4. It is assumed the system is able to produce a list of all vaccines’ distributed quantities. The system is assumed to be able to allow the employees to list all distributed vaccines and their accumulated quantities that was recorded in the dist.txt file. These distributed quantities will have to be sorted descendingly (with highest quantity first and lowest quantity last).

3. Design of the Program

3.1 Design - Pseudocode

0. Structs.

```
DECLARE struct Vaccine{
    DECLARE character variable name, code, country
    SET name to size 30
    SET country to size 30
    SET code to size 4
    DECLARE a int variable dosage
    DECLARE float variable population,    quantity
    }
DECLARE struct Distributed{
    DECLARE character variable code, status
    SET code to size 4
    SET status to size 12
    DECLARE a float variable quantity
    }
DECLARE struct distinfo{
    DECLARE a character variable code
    SET code to size 4
    DECLARE a float variable quantity
}
```

1. Menu

```
BEGIN
    DISPLAY "Vaccine Inventory Management System created by Macel"
    WHILE True
    DISPLAY "Select a Function :
    DISPLAY "1. Create New inventory"
    DISPLAY "2. Update Vaccine Quantity"
    DISPLAY "3. Search Vaccine"
    DISPLAY "4. View List of all Vaccines and their Distributed Quantities"
    DISPLAY "5. Exit Program"
        READ AS userinput
        WHILE userinput is more than 5 OR userinput is less than 0
            DISPLAY "Invalid Input, Please Select a Function : "
            DISPLAY "1. Create New inventory"
            DISPLAY "2. Update Vaccine Quantity"
            DISPLAY "3. Search Vaccine"
            DISPLAY "4. View List of all Vaccines and their Distributed Quantities"
            DISPLAY "5. Exit Program"
            READ AS userinput
        ENDWHILE
        CASE based on userinput
            CASE '1'
                CALL CreateInventory
            CASE '2'
                CALL UpdateVaccineQnty
            CASE '3'
                CALL SearchVaccine
            CASE '4'
                CALL ViewVacDist
            CASE '5'
                DISPLAY "Exiting Program"
                CALL exit
            DEFAULT
        ENDCASE
    ENDWHILE
END
```

2. Main Function : Inventory Creation

```
CREATE FUNCTION createInventory
    DECLARE integer variable confirm, DosRqrd
    DECLARE float variable PopCover, VacQnty
    SET confirm to 69
    DECLARE a character array NmOfVac, SET NmOfVac to size 30
    DECLARE a character array VacCode, SET VacCode to size 4
    DECLARE a character array ProdCntr, SET ProdCntr to size 30
    OPENFILE "vaccine.txt" FOR APPEND
    DO WHILE confirm not equal to 1
        DISPLAY "Creating New Vaccine Inventory, Enter 1 to Continue, or 0 to go Back : "
        SCAN confirm
        IF confirm equal to 0
            RETURN to main menu
        ENDIF
    ENDDO
    DO WHILE confirm not equal to 1
        DISPLAY "Enter Name of Vaccine              : " GET NmOfVac
        DISPLAY "Enter Vaccine Code                 : " GET VacCode
        DISPLAY "Enter Producing Country            : " GET ProdCntr
        DISPLAY "Dosage Required                    : " SCAN DosRqrd
        DISPLAY "Population Covered                 : " SCAN PopCover
        DISPLAY "Vaccine Quantity (In Mill)         : " SCAN VacQnty
        SET confirm equal to 69
        DISPLAY "Enter 2 to Save and Create another inventory"
        DISPLAY "1 to Save and go back to Main Menu"
        DISPLAY "0 to Discard and go back to Main Menu"
        DISPLAY "or Any Number to Redo"
        SCAN confirm
        IF confirm equal to 0
            CLOSEFILE "vaccine.txt"
```

```
                RETURN to main menu
            ELSE IF confirm equal to 1 OR confirm equal to 2
                DISPLAY "Vaccine Details Successfully Saved"
                WRITEFILE "vaccine.txt", "NmOfVac, VacCode, ProdCntr, DosRqrd,
PopCover, VacQnty"
            ENDIF
        ENDDO
        CLOSEFILE "vaccine.txt"
        RETURN to main menu
    END FUNCTION
```

3. Main Function : Update Vaccine Quantities

```
CREATE FUNCTION updateVaccineQnty
    DECLARE a character array userinput, SET userinput to size 4
    DECLARE a float variable qntyinput
    DECLARE integer variable i, confirm, count, counter
    SET counter to 1
    CALL countLines(count, "vaccine.txt")
    DECLARE a vaccine struct array called vac, SET vac to size count
    DISPLAY "Updating Vaccine Quantity"
    DO WHILE confirm not equal to 1
        DISPLAY "Enter 1 to Continue, or 0 to go back to Main Menu : "
        READ confirm
        IF confirm equal to 0
            RETURN to main menu
        ENDIF
    ENDDO
    DISPLAY "Listing Vaccine Names with each Codes and Quantities (In Mill) : "
    OPENFILE "vaccine.txt" FOR READ
    FOR i = 0 to i less than count
        READFILE "vaccine.txt", "vac[i].name, vac[i].code, vac[i].country, vac[i].dosage,
vac[i].population, vac[i].quantity"
        DISPLAY "vac[i].name = vac[i].code, vac[i].quantity"
    ENDFOR
    CLOSEFILE "vaccine.txt"
    DISPLAY "Select a Vaccine code and modify the quantity : ")
    WHILE counter equal to 1
        SCAN userinput
        FOR i = 0 to i less than count
            IF userinput equal to vac[i].code
                SET counter to 0
                END FOR
            ELSE IF i equal to count-1
                DISPLAY "Vaccine Code not Found, Please Enter Again : "
            ENDIF
        ENDFOR
    ENDWHILE
    DISPLAY "vac.name, Quantity : vac[i].quantity"
    DISPLAY "Enter 1 to Add Quantity, or 0 to Distribute Quantity : "
    SCAN confirm
    IF confirm equal to 1
        DISPLAY "Quantity to be Added (In Million) : "
        SCAN qntyinput
        ADD vac[i].quantity with vac[i].quantity and qntyinput
        OPENFILE "vaccine.txt" FOR WRITE
        OPENFILE "dist.txt" FOR APPEND
        WRITEFILE "dist.txt", "vac[i].code Added qntyinput"
        FOR i = 0 to i less than count
            WRITEFILE "vaccine.txt", "vac[i].name, vac[i].code, vac[i].country,
vac[i].dosage, vac[i].population, vac[i].quantity"
        END FOR
        CLOSEFILE "vaccine.txt"
        CLOSEFILE "dist.txt"
    ELSE IF confirm equal to 0
        DISPLAY "Quantity to be Distributed (In Million) : "
        SCAN qntyinput
        SUBTRACT vac.quantity with qntyinput
        WHILE vac[i].quantity less than 0
            DISPLAY "Not Enough Vaccine to be Distributed"
            ADD vac[i].quantity with qntyinput
            DISPLAY "Please Re-Enter Distributed Quantity (In Million) : "
            SCAN qntyinput
            SUBTRACT vac[i].quantity with qntyinput
        ENDWHILE
        DISPLAY "Successfully Updated Vaccine Quantity"
        DISPLAY "Updated Vaccine Details : "
        DISPLAY "vac[i].name = vac[i].code, vac[i].quantity"
        OPENFILE "vaccine.txt" FOR WRITE
        OPENFILE "dist.txt" FOR APPEND
        WRITEFILE "dist.txt", "vac.code Distributed qntyinput"
        FOR i = 0 to i less than count
            WRITEFILE "vaccine.txt", "vac[i].name, vac[i].code, vac[i].country,
vac[i].dosage, vac[i].population, vac[i].quantity"
        ENDFOR
        CLOSEFILE "vaccine.txt"
        CLOSEFILE "dist.txt"
    ELSE
        DISPLAY "Invalid Input, Returning to Main Menu"
        RETURN to main menu
    ENDIF
    DISPLAY "Returning to Main Menu"
    RETURN to main menu
END FUNCTION
```

4. Main Function : Vaccine Search by using Vaccine Code

```
CREATE FUNCTION searchVaccine
    DECLARE a character array userinput, SET userinput to size 4
    DECLARE integer variable confirm, i, counter, count
    CALL countLines(count, "vaccine.txt")
    DECLARE a vaccine struct array called vac, SET vac to size count
    DISPLAY "Search Vaccine Quantity by Vaccine Code"
    DO WHILE confirm not equal to 1
        DISPLAY "Enter 1 to Continue, or 0 to go back to Main Menu : "
        SCAN confirm
        IF confirm equal to 0
            RETURN to main menu
        ENDIF
    ENDDO
    OPENFILE "vaccine.txt" FOR READ
    DISPLAY "Enter Vaccine Code : "
    WHILE confirm equal to 1
        SCAN userinput
        FOR i = 0 to i less than count
            READFILE "vaccine.txt", "vac[i].name, vac[i].code, vac[i].country,
va[i]c.dosage, vac[i].population, vac[i].quantity"
            IF userinput equal to vac[i].code
                SET confirm to 0
                ENDFOR
            ELSE IF i equal to count-1
                DISPLAY "Vaccine Code not Found, Please Enter Again : "
                CLOSEFILE "vaccine.txt"
            ENDIF
        ENDFOR
    ENDWHILE
    DISPLAY "Vaccine Found"
```

```
        DISPLAY "Displaying Vaccine Details : "
        DISPLAY "Vaccine Name : vac[i].name"
        DISPLAY "Vaccine Code : vac[i].code"
        DISPLAY "Vaccine Country : vac[i].country"
        DISPLAY "Vaccine Dosage Needed : vac[i].dosage"
        DISPLAY "Vaccine Population Coverage : vac[i].population"
        DISPLAY "Vaccine Quantity (In Mill) : vac[i].quantity"
        DISPLAY "Returning to Main Menu"
        CLOSEFILE "vaccine.txt"
        RETURN to main menu
    END FUNCTION
```

5. Main Function : Viewing Total Distributed Vaccine Quantity for each Vaccine Code

```
CREATE FUNCTION viewVacDist
    DECLARE character array temp, SET temp size to 4
    DECLARE integer variable i, j, count, vaccount
    CALL countLines(count, "dist.txt")
    DECLARE a Distributed struct array called dist, SET dist to size count
    DECLARe a distinfo struct array called dinfo, SET dinfo to size count
    CALL countLines(vaccount, "vaccine.txt")
    DECLARE a Vaccine struct array called vac, SET vac to size vaccount
    DISPLAY "Viewing List of all Vaccines and their Distributed Quantities"
    OPENFILE "dist.txt" FOR READ
    OPENFILE "vaccine.txt" FOR READ
    FOR i = 0 to i less than vaccount
        READFILE "vaccine.txt", "vac[i].name, vac[i].code, vac[i].country, vac[i].dosage,
vac[i].population, vac[i].quantity"
    ENDFOR
    FOR i = 0 to i less than count
        READFILE "dist.txt", dist[i].code, dist[i].status, dist[i].quantity
    ENDFOR
    FOR i = 0 to i less than vaccount
        STRING COPY vac[i].code to dinfo[i].code
        FOR j = 0 to j less than count
            IF dist[j].code is equal to vac[i].code AND dist[j].status equal to "Distributed"
                ADD dinfo[i].quantity with dist[j].quantity
            ENDIF
        ENDFOR
    ENDFOR
    FOR i=0 to i less than count
        FOR j = 0 to j less than vaccount-i-1
            IF dinfo[i].quantity less than dinfo[i+1].quantity
                CALL swapFloat(dinfo[i].quantity, dinfo[i+1].quantity)
                STRING COPY dinfo[j].code to temp
                STRING COPY dinfo[j+1].code to dinfo[j].code
                STRING COYP temp to dinfo[j+1].code
            ENDIF
        ENDFOR
    ENDFOR      DISPLAY "Listing Vaccine code with Distrubuted Value Descendingly (From
Highest Quantity to Lowest) : "
    FOR i = 0 to i less than vaccount
        DISPLAY "(dinfo[i].code) Total Distributed : dinfo[i].quantity Million"
    ENDFOR
    RETURN to main menu
END FUNCTION
```

6. Extra Functions : Swap Float

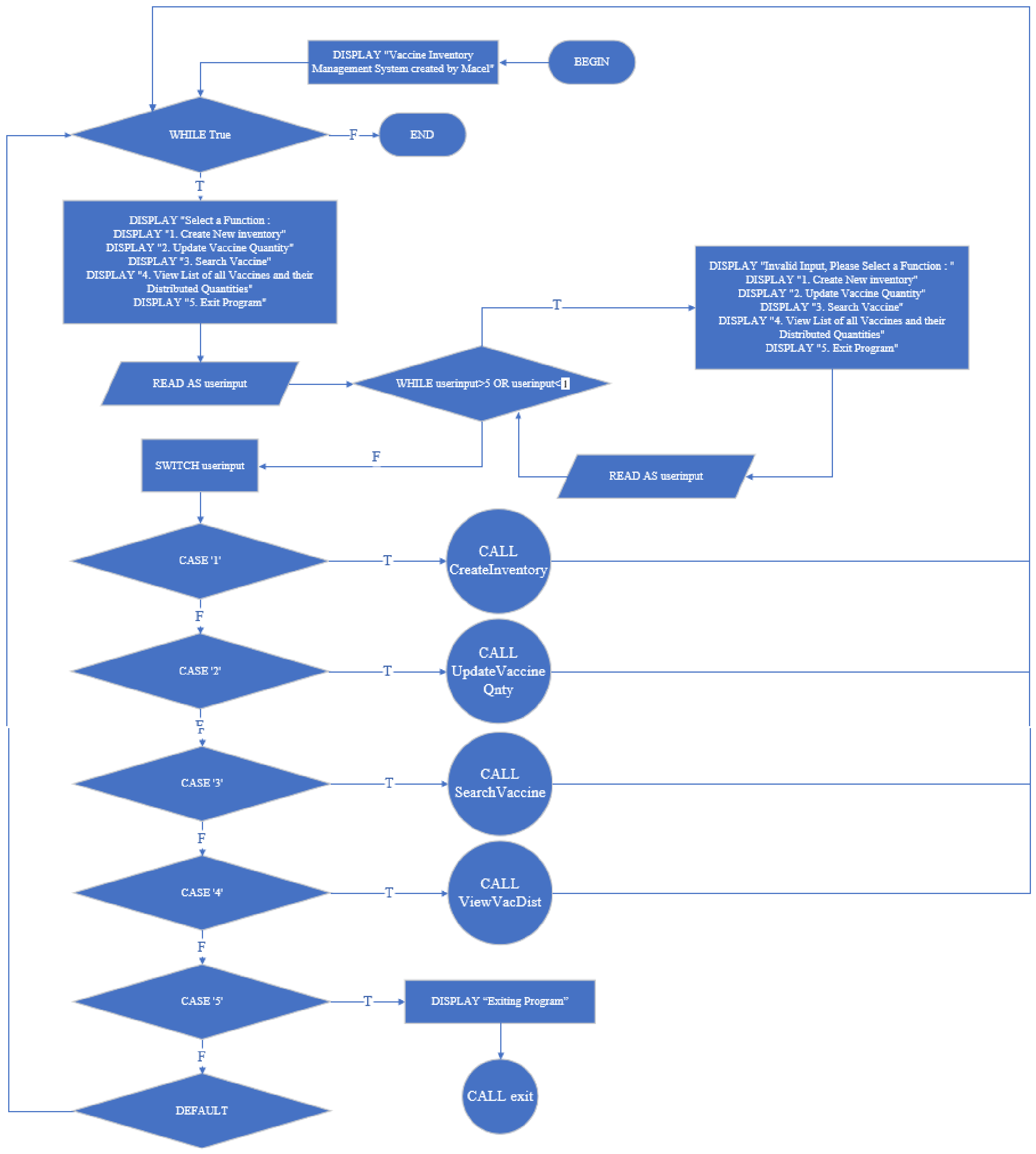
```
CREATE FUNCTION swapFloat(float* xp, float* yp)
    DECLARE a float variable temp
    SET temp to *xp
    SET *xp to *yp
    SET *yp to temp
    RETURN
END FUNCTION
```

7. Extra Functions : Count Lines

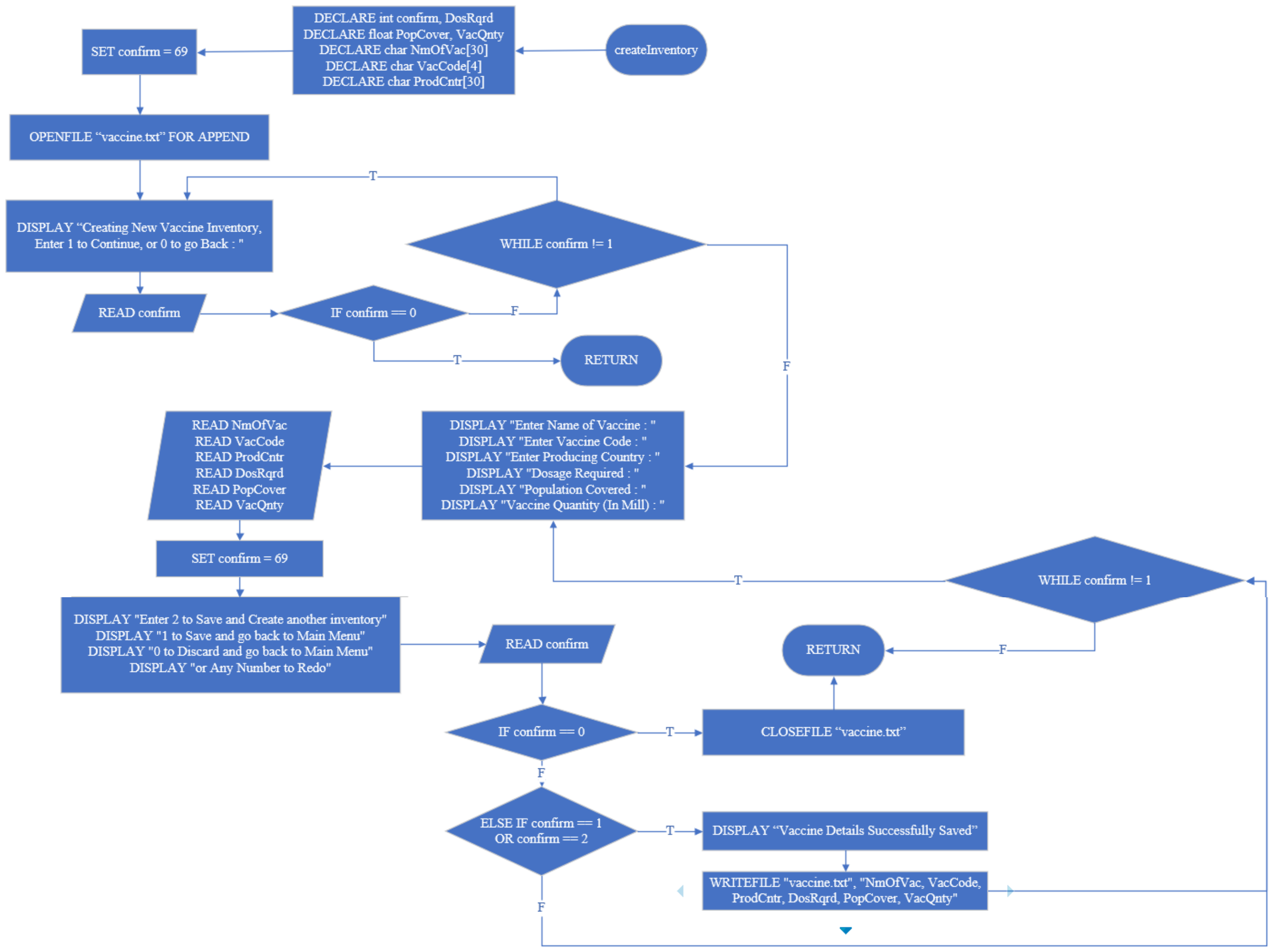
```
CREATE FUNCTION countLines(char* fileinput)
    DECLARE integer variable lines, count
    SET lines to 0
    SET count to 0
    OPENFILE "fileinput" FOR READ
    IF "fileinput" equal to NULL
        DISPLAY "File Not Found, Exiting Program"
        CALL exit
    ENDIF
    FOR lines in "fineinput" to lines not equal to EOF
        IF lines equal to next lines
            ADD 1 to count
        ENDIF
    ENDFOR
    CLOSFILE "fileinput"
    RETURN count
END FUNCTION
```

3.2 Design – Flowchart

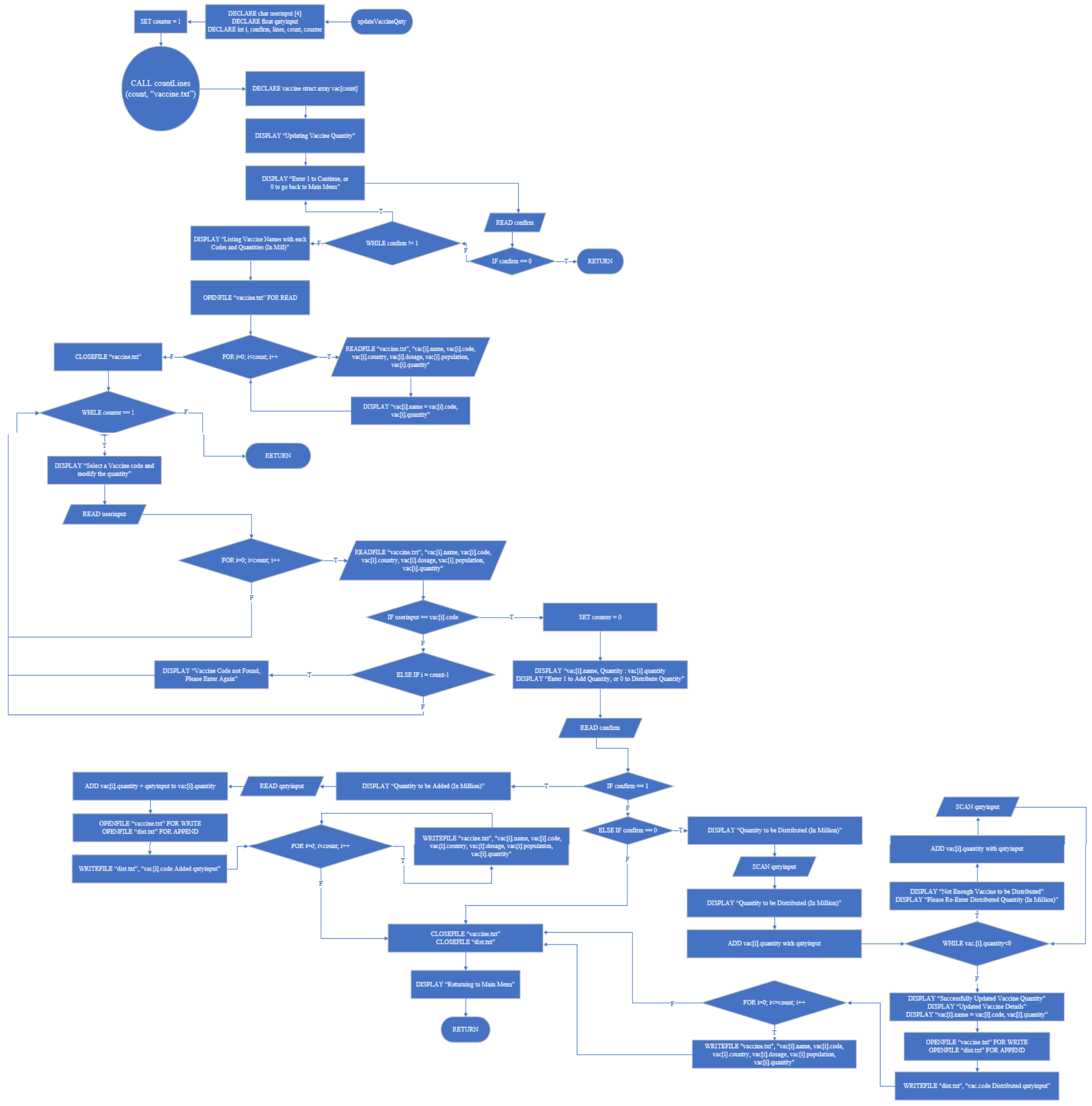
1. Main Menu


BEGIN
DISPLAY "Vaccine Inventory Management System created by Macel"
WHILE True
F
END
T
DISPLAY "Select a Function :
DISPLAY "1. Create New inventory"
DISPLAY "2. Update Vaccine Quantity"
DISPLAY "3. Search Vaccine"
DISPLAY "4. View List of all Vaccines and their Distributed Quantities"
DISPLAY "5. Exit Program"
READ AS userinput
WHILE userinput>5 OR userinput<1
T
DISPLAY "Invalid Input, Please Select a Function : "
DISPLAY "1. Create New inventory"
DISPLAY "2. Update Vaccine Quantity"
DISPLAY "3. Search Vaccine"
DISPLAY "4. View List of all Vaccines and their Distributed Quantities"
DISPLAY "5. Exit Program"
READ AS userinput
F
SWITCH userinput
CASE '1'
T
CALL CreateInventory
F
CASE '2'
T
CALL UpdateVaccine Qnty
F
CASE '3'
T
CALL SearchVaccine
F
CASE '4'
T
CALL ViewVacDist
F
CASE '5'
T
DISPLAY "Exiting Program"
CALL exit
F
DEFAULT

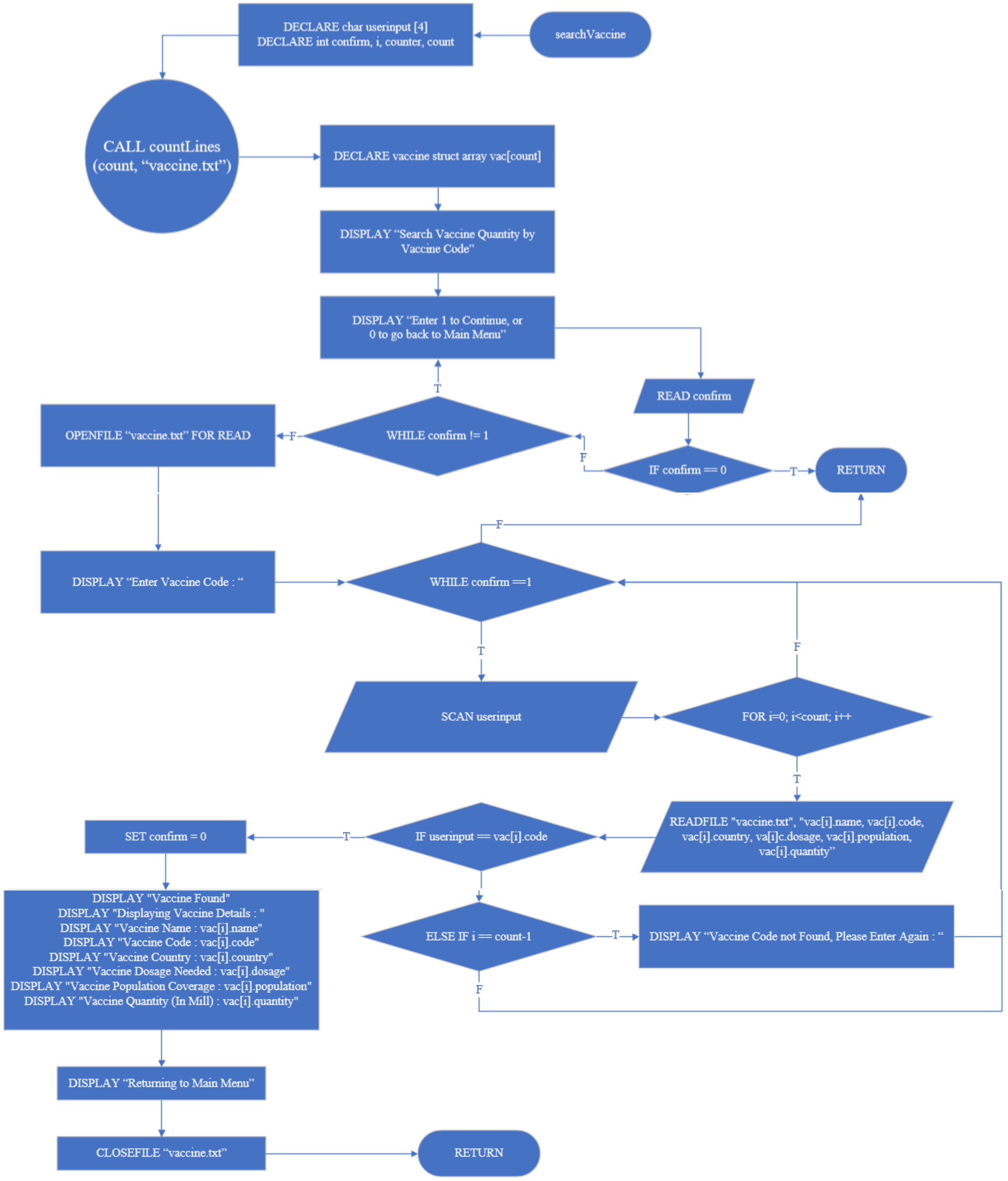
2. Inventory Creation


createInventory
DECLARE int confirm, DosRqrd
DECLARE float PopCover, VacQnty
DECLARE char NmOfVac[30]
DECLARE char VacCode[4]
DECLARE char ProdCntr[30]
SET confirm = 69
OPENFILE "vaccine.txt" FOR APPEND
DISPLAY "Creating New Vaccine Inventory, Enter 1 to Continue, or 0 to go Back : "
READ confirm
IF confirm == 0
RETURN
WHILE confirm != 1
T
F
DISPLAY "Enter Name of Vaccine : "
DISPLAY "Enter Vaccine Code : "
DISPLAY "Enter Producing Country : "
DISPLAY "Dosage Required : "
DISPLAY "Population Covered : "
DISPLAY "Vaccine Quantity (In Mill) : "
READ NmOfVac
READ VacCode
READ ProdCntr
READ DosRqrd
READ PopCover
READ VacQnty
SET confirm = 69
DISPLAY "Enter 2 to Save and Create another inventory"
DISPLAY "1 to Save and go back to Main Menu"
DISPLAY "0 to Discard and go back to Main Menu"
DISPLAY "or Any Number to Redo"
READ confirm
WHILE confirm != 1
RETURN
IF confirm == 0
CLOSEFILE "vaccine.txt"
ELSE IF confirm == 1 OR confirm == 2
DISPLAY "Vaccine Details Successfully Saved"
WRITEFILE "vaccine.txt", "NmOfVac, VacCode, ProdCntr, DosRqrd, PopCover, VacQnty"

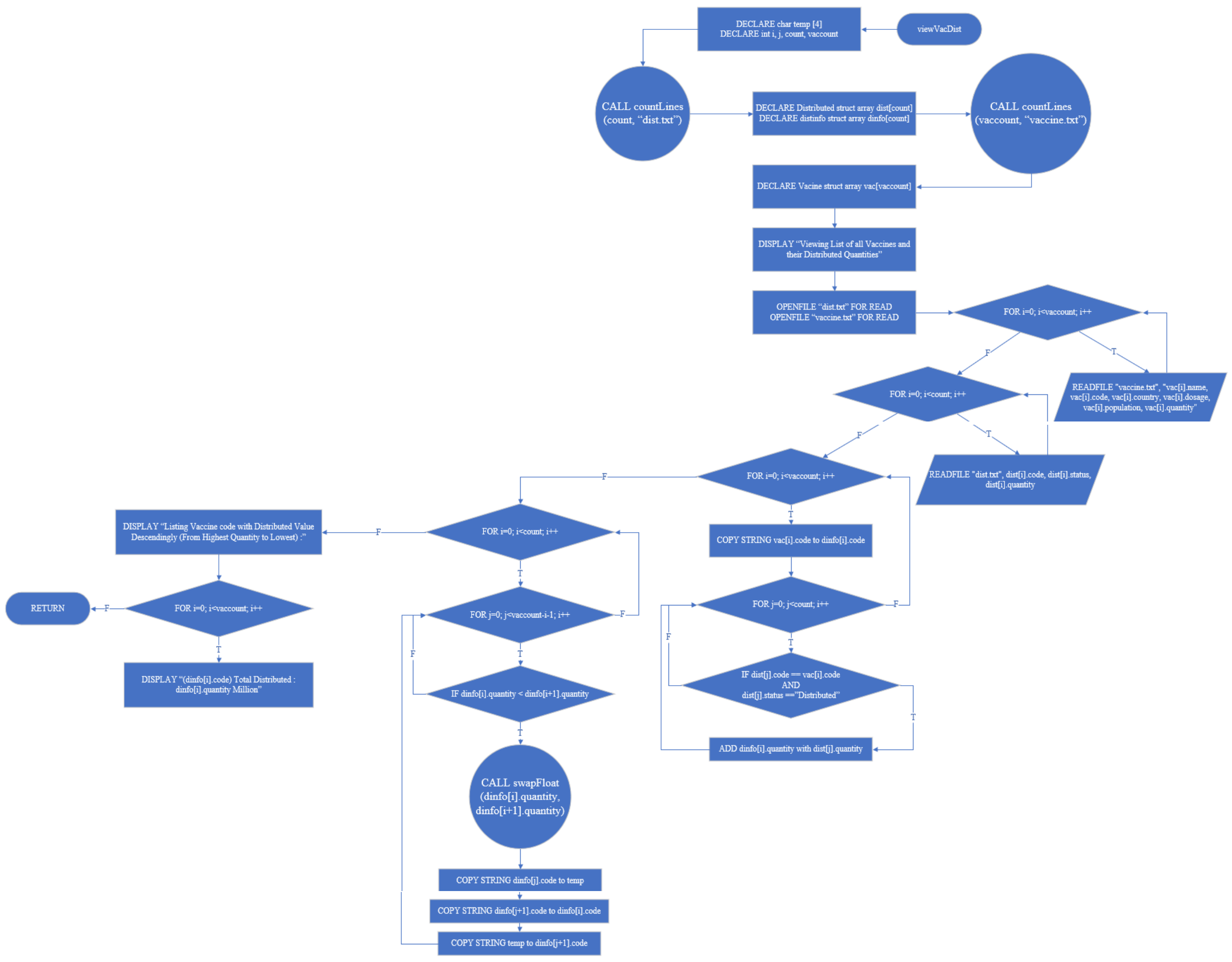
3. Update Vaccine Quantities


updateVaccineQnty
DECLARE char userinput [4]
DECLARE float qntyinput
DECLARE int i, confirm, lines, count, counter
SET counter = 1
CALL countLines (count, "vaccine.txt")
DECLARE vaccine struct array vac[count]
DISPLAY "Updating Vaccine Quantity"
DISPLAY "Enter 1 to Continue, or 0 to go back to Main Menu"
READ confirm
IF confirm == 0
T
RETURN
F
WHILE confirm != 1
T
F
DISPLAY "Listing Vaccine Names with each Codes and Quantities (In Mill)"
OPENFILE "vaccine.txt" FOR READ
FOR i=0; i<count; i++
T
READFILE "vaccine.txt", "vac[i].name, vac[i].code, vac[i].country, vac[i].dosage, vac[i].population, vac[i].quantity"
DISPLAY "vac[i].name = vac[i].code, vac[i].quantity"
F
CLOSEFILE "vaccine.txt"
WHILE counter == 1
F
RETURN
T
DISPLAY "Select a Vaccine code and modify the quantity"
READ userinput
FOR i=0; i<count; i++
T
READFILE "vaccine.txt", "vac[i].name, vac[i].code, vac[i].country, vac[i].dosage, vac[i].population, vac[i].quantity"
F
IF userinput == vac[i].code
T
SET counter = 0
F
ELSE IF i = count-1
T
DISPLAY "Vaccine Code not Found, Please Enter Again"
F
DISPLAY "vac[i].name, Quantity : vac[i].quantity
DISPLAY "Enter 1 to Add Quantity, or 0 to Distribute Quantity"
READ confirm
IF confirm == 1
T
DISPLAY "Quantity to be Added (In Million)"
READ qntyinput
ADD vac[i].quantity + qntyinput to vac[i].quantity
OPENFILE "vaccine.txt" FOR WRITE
OPENFILE "dist.txt" FOR APPEND
WRITEFILE "dist.txt", "vac[i].code Added qntyinput"
FOR i=0; i<count; i++
T
WRITEFILE "vaccine.txt", "vac[i].name, vac[i].code, vac[i].country, vac[i].dosage, vac[i].population, vac[i].quantity"
F
F
ELSE IF confirm == 0
T
DISPLAY "Quantity to be Distributed (In Million)"
SCAN qntyinput
DISPLAY "Quantity to be Distributed (In Million)"
ADD vac[i].quantity with qntyinput
WHILE vac.[i].quantity<0
T
DISPLAY "Not Enough Vaccine to be Distributed"
DISPLAY "Please Re-Enter Distributed Quantity (In Million)"
ADD vac[i].quantity with qntyinput
SCAN qntyinput
F
DISPLAY "Successfully Updated Vaccine Quantity"
DISPLAY "Updated Vaccine Details"
DISPLAY "vac[i].name = vac[i].code, vac[i].quantity"
OPENFILE "vaccine.txt" FOR WRITE
OPENFILE "dist.txt" FOR APPEND
WRITEFILE "dist.txt", "vac.code Distributed qntyinput"
FOR i=0; i<=count; i++
T
WRITEFILE "vaccine.txt", "vac[i].name, vac[i].code, vac[i].country, vac[i].dosage, vac[i].population, vac[i].quantity"
F
F
CLOSEFILE "vaccine.txt"
CLOSEFILE "dist.txt"
DISPLAY "Returning to Main Menu"
RETURN

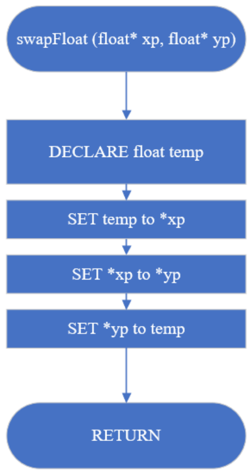
4. Vaccine Search by using Vaccine Code


searchVaccine
DECLARE char userinput [4]
DECLARE int confirm, i, counter, count
CALL countLines (count, "vaccine.txt")
DECLARE vaccine struct array vac[count]
DISPLAY "Search Vaccine Quantity by Vaccine Code"
DISPLAY "Enter 1 to Continue, or 0 to go back to Main Menu"
READ confirm
IF confirm == 0
T
RETURN
F
WHILE confirm != 1
T
F
OPENFILE "vaccine.txt" FOR READ
DISPLAY "Enter Vaccine Code : "
WHILE confirm ==1
F
T
SCAN userinput
FOR i=0; i<count; i++
F
T
READFILE "vaccine.txt", "vac[i].name, vac[i].code, vac[i].country, va[i]c.dosage, vac[i].population, vac[i].quantity"
IF userinput == vac[i].code
T
SET confirm = 0
DISPLAY "Vaccine Found"
DISPLAY "Displaying Vaccine Details : "
DISPLAY "Vaccine Name : vac[i].name"
DISPLAY "Vaccine Code : vac[i].code"
DISPLAY "Vaccine Country : vac[i].country"
DISPLAY "Vaccine Dosage Needed : vac[i].dosage"
DISPLAY "Vaccine Population Coverage : vac[i].population"
DISPLAY "Vaccine Quantity (In Mill) : vac[i].quantity"
ELSE IF i == count-1
T
DISPLAY "Vaccine Code not Found, Please Enter Again : "
F
DISPLAY "Returning to Main Menu"
CLOSEFILE "vaccine.txt"
RETURN

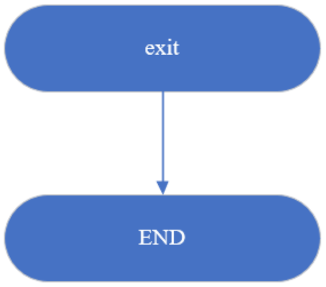
5. Viewing Total Distributed Vaccine Quantity for each Vaccine Code


viewVacDist
DECLARE char temp [4]
DECLARE int i, j, count, vaccount
CALL countLines (count, "dist.txt")
DECLARE Distributed struct array dist[count]
DECLARE distinfo struct array dinfo[count]
CALL countLines (vaccount, "vaccine.txt")
DECLARE Vacine struct array vac[vaccount]
DISPLAY "Viewing List of all Vaccines and their Distributed Quantities"
OPENFILE "dist.txt" FOR READ
OPENFILE "vaccine.txt" FOR READ
FOR i=0; i<vaccount; i++
T
READFILE "vaccine.txt", "vac[i].name, vac[i].code, vac[i].country, vac[i].dosage, vac[i].population, vac[i].quantity"
F
FOR i=0; i<count; i++
T
READFILE "dist.txt", dist[i].code, dist[i].status, dist[i].quantity
F
FOR i=0; i<vaccount; i++
T
COPY STRING vac[i].code to dinfo[i].code
FOR j=0; j<count; i++
T
IF dist[j].code == vac[i].code AND dist[j].status =="Distributed"
T
ADD dinfo[i].quantity with dist[j].quantity
F
F
F
FOR i=0; i<count; i++
T
FOR j=0; j<vaccount-i-1; i++
T
IF dinfo[i].quantity < dinfo[i+1].quantity
T
CALL swapFloat (dinfo[i].quantity, dinfo[i+1].quantity)
COPY STRING dinfo[j].code to temp
COPY STRING dinfo[j+1].code to dinfo[i].code
COPY STRING temp to dinfo[j+1].code
F
F
F
DISPLAY "Listing Vaccine code with Distributed Value Descendingly (From Highest Quantity to Lowest) :"
FOR i=0; i<vaccount; i++
T
DISPLAY "(dinfo[i].code) Total Distributed : dinfo[i].quantity Million"
F
RETURN

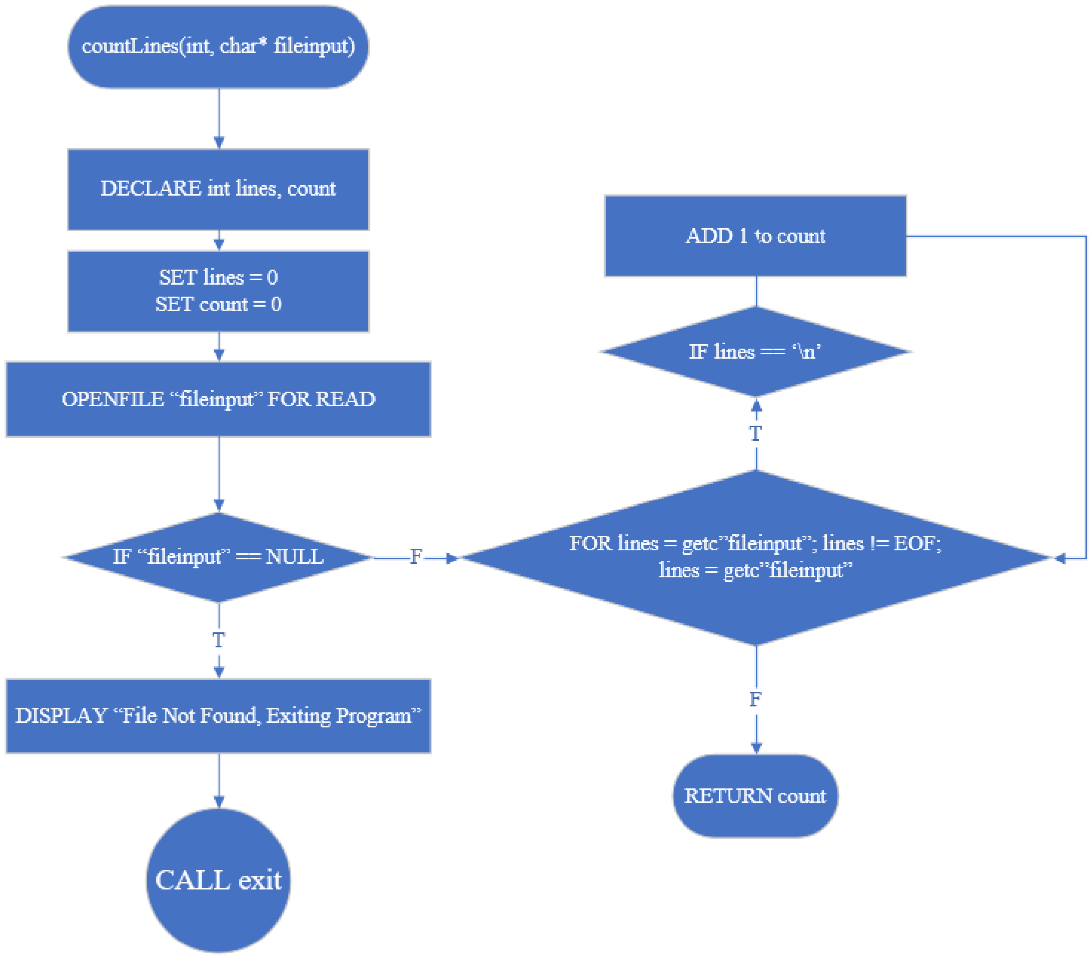
6. Swap Float



7. Exit



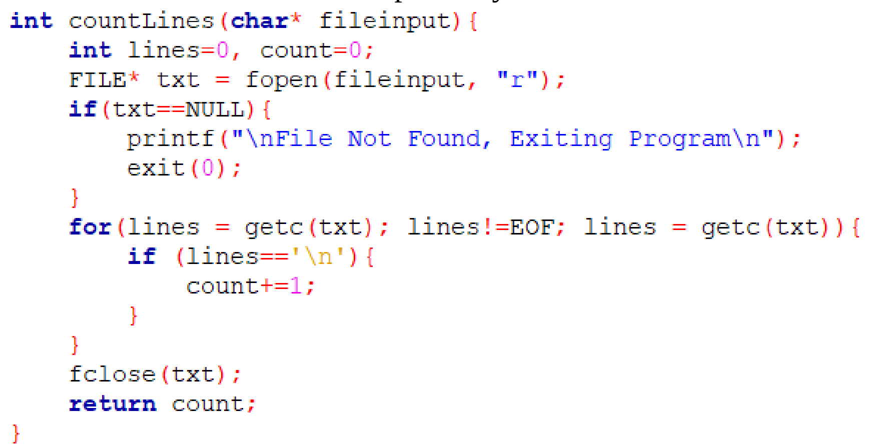
8. Count Lines



4. Additional Features

1. Flexibility

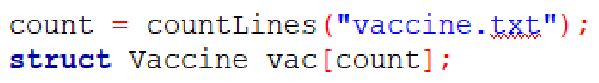
   The program are able to update all functions if there's new vaccine. In a function to view all vaccines, generally, for loop is used to show all the data in the txt file, but the for loop is usually made to loop only a static number of times (for ex. if there are 5 vaccines, the for loop only loops 5 times), but with this program, a function called countLines is implemented to count how many lines there is in the txt file and the for loop will loop for the number of lines in the txt file, even if the employee adds another line in the txt file, other functions will still work perfectly.



```
int countLines(char* fileinput){
    int lines=0, count=0;
    FILE* txt = fopen(fileinput, "r");
    if(txt==NULL){
        printf("\nFile Not Found, Exiting Program\n");
        exit(0);
    }
    for(lines = getc(txt); lines!=EOF; lines = getc(txt)){
        if (lines=='\n'){
            count+=1;
        }
    }
    fclose(txt);
    return count;
}
```

2. Memory Saving

   Since the program uses struct array, which needs the size to be specified, general programs will set a specific amount of size to the struct array, which could lead to struct array size too big which wastes memory, or struct array size too small, which will make the program not work, by using the countLines function to count the lines of a txt file, the size of the struct array will match the number of lines in the txt file, making it very efficient.



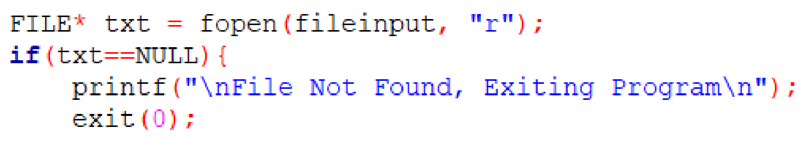
```
count = countLines("vaccine.txt");
struct Vaccine vac[count];
```

3. Can be easily upgraded

   Since the program is not one big chunk of files, but rather divided into multiple files, the program is easy to be upgraded, if a new function were to be added, the programmer/ developer will have an easier time working on the program, since multiple files are easier to read, maintain and manage.

4. File validation functions.

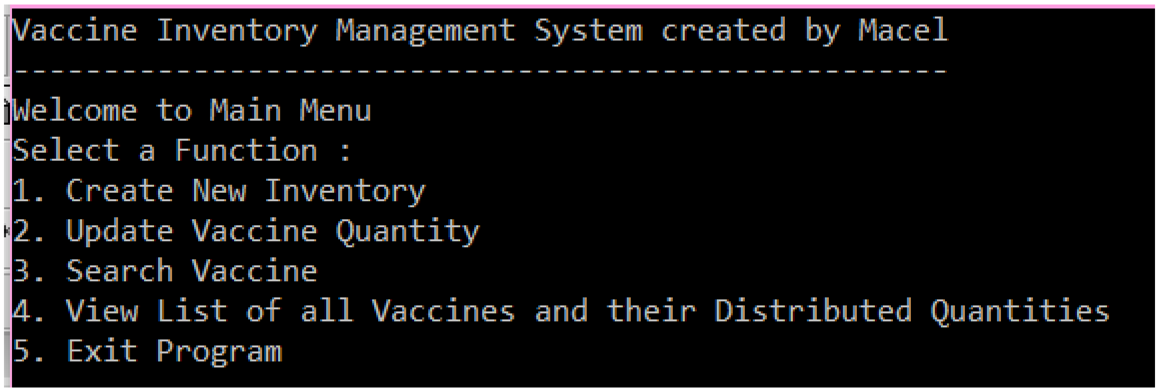
   The program also has a function which checks if the file exists or not, if the file was not found (the file might be accidentally deleted), the program will display "File Not Found" message, and the program will be closed. This is to ensure the user doesn't get random information if the file can't be accessed.



```
FILE* txt = fopen(fileinput, "r");
if(txt==NULL){
    printf("\nFile Not Found, Exiting Program\n");
    exit(0);
```

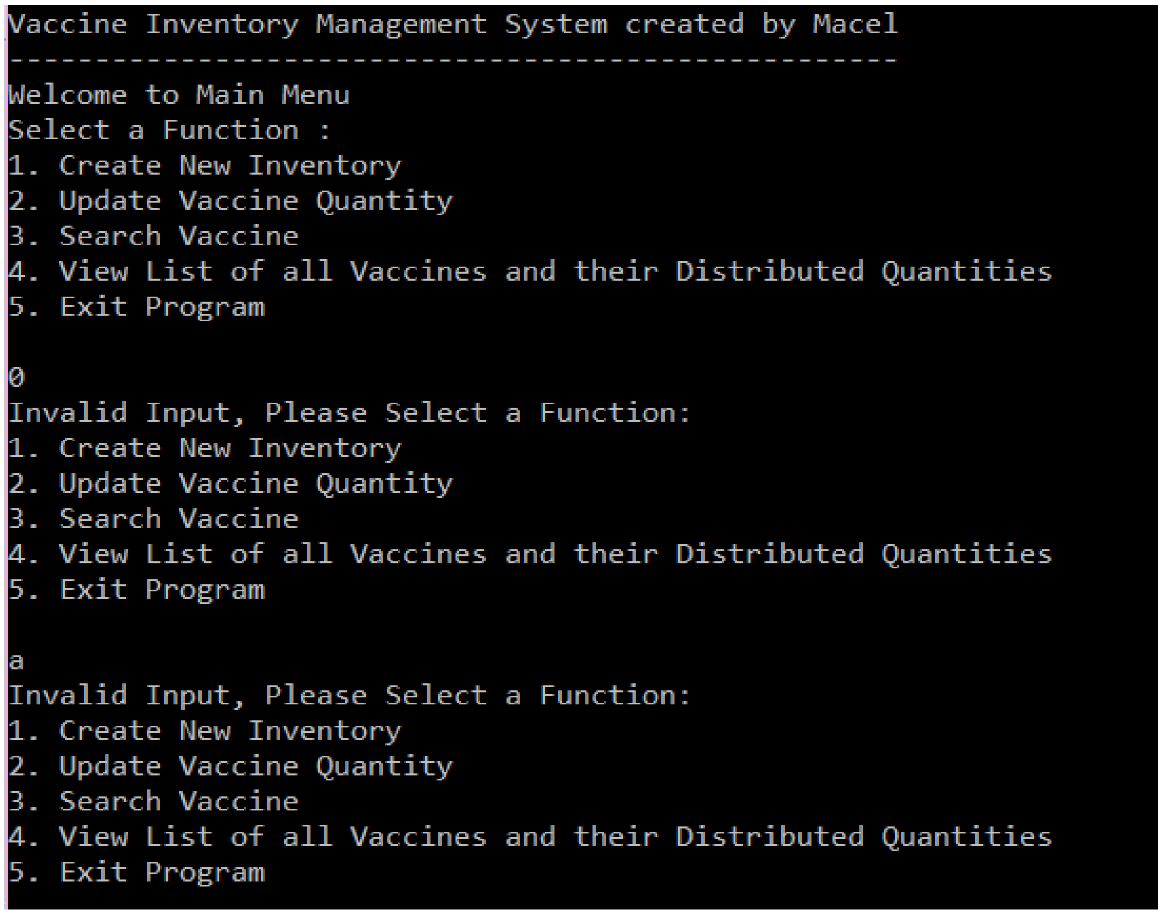
5. Screenshot of Input/Output

Right after executing the program, the program name will be showed, as well as a welcome to main menu message, the program asks the user to input 1-5 based on the desired function the user wants to execute.



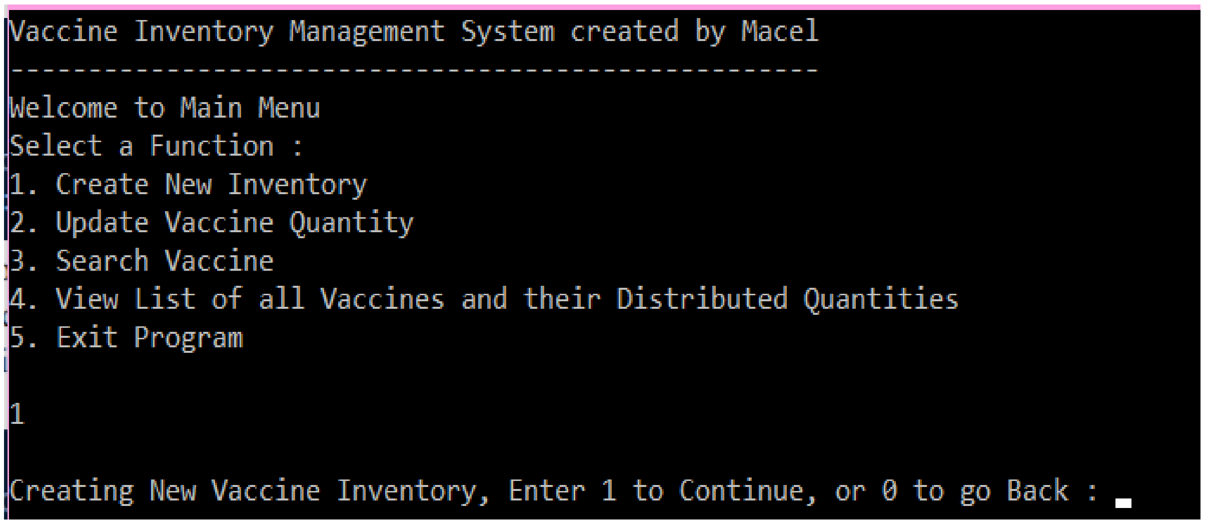
```
Vaccine Inventory Management System created by Macel
-----------------------------------------------------
Welcome to Main Menu
Select a Function :
1. Create New Inventory
2. Update Vaccine Quantity
3. Search Vaccine
4. View List of all Vaccines and their Distributed Quantities
5. Exit Program
```

If the user inputs anything else than 1-5, the program will display "Invalid Input" message and it will ask the user to select the right input.



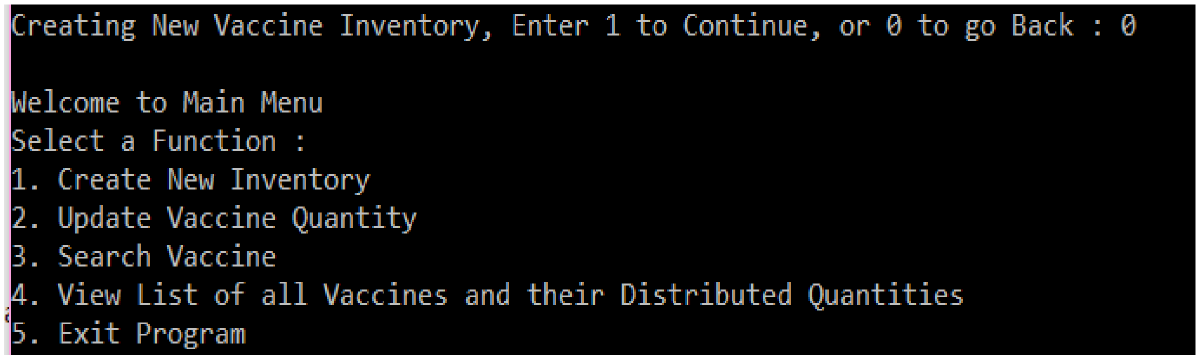
```
Vaccine Inventory Management System created by Macel
-----------------------------------------------------
Welcome to Main Menu
Select a Function :
1. Create New Inventory
2. Update Vaccine Quantity
3. Search Vaccine
4. View List of all Vaccines and their Distributed Quantities
5. Exit Program

0
Invalid Input, Please Select a Function:
1. Create New Inventory
2. Update Vaccine Quantity
3. Search Vaccine
4. View List of all Vaccines and their Distributed Quantities
5. Exit Program

a
Invalid Input, Please Select a Function:
1. Create New Inventory
2. Update Vaccine Quantity
3. Search Vaccine
4. View List of all Vaccines and their Distributed Quantities
5. Exit Program
```

The first function is Create New Inventory, after the user selects the function, a verification will pop up asking if the user wants to continue or go back to main menu, this feature is for the sake of user convenience, in case the user inputs the wrong function other than the one they originally intended



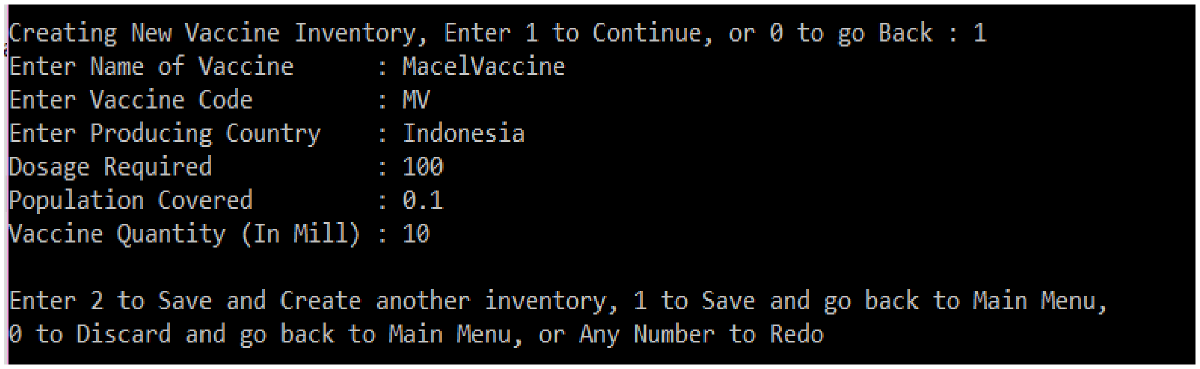
```
Vaccine Inventory Management System created by Macel
----------------------------------------------------
Welcome to Main Menu
Select a Function :
1. Create New Inventory
2. Update Vaccine Quantity
3. Search Vaccine
4. View List of all Vaccines and their Distributed Quantities
5. Exit Program

1

Creating New Vaccine Inventory, Enter 1 to Continue, or 0 to go Back :
```

As mentioned, entering 0 will return the user back to main menu and asks them to again choose the desired function



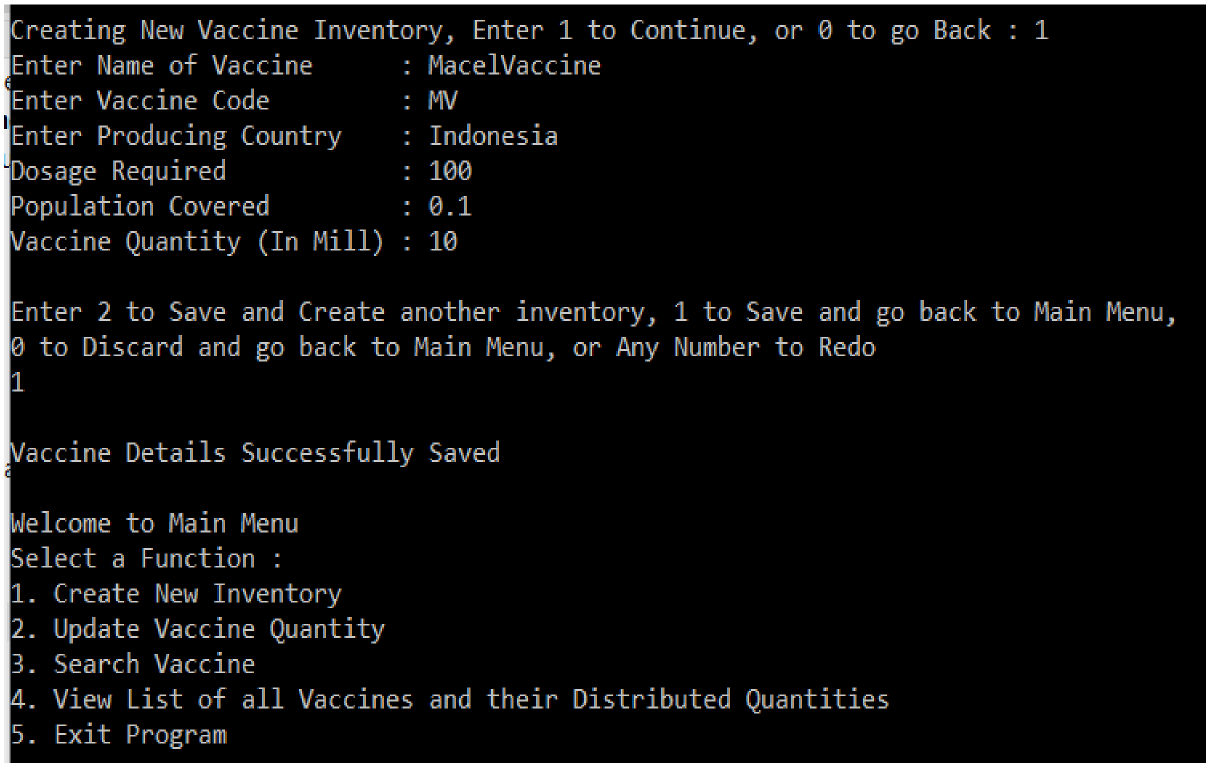
```
Creating New Vaccine Inventory, Enter 1 to Continue, or 0 to go Back : 0

Welcome to Main Menu
Select a Function :
1. Create New Inventory
2. Update Vaccine Quantity
3. Search Vaccine
4. View List of all Vaccines and their Distributed Quantities
5. Exit Program
```

After the user selects 1 from the verify feature, the user then will be asked to input vaccine details that will be added into the vaccine.txt file. The user will also be asked if they want to enter 2 to save and create another inventory (in case there are more than one new vaccine), 1 to save and go back to main menu, 0 to discard and go back to main menu, or any number to redo the inputs (incase the user inputs the wrong detail)



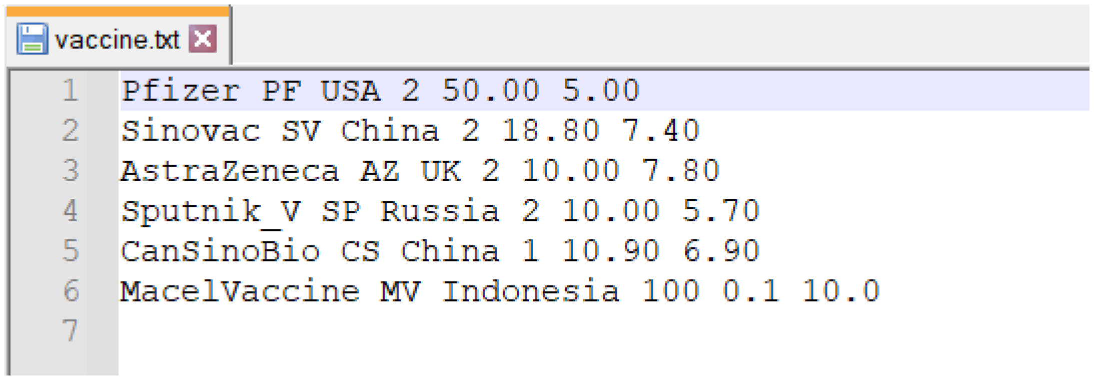
```
Creating New Vaccine Inventory, Enter 1 to Continue, or 0 to go Back : 1
Enter Name of Vaccine      : MacelVaccine
Enter Vaccine Code         : MV
Enter Producing Country    : Indonesia
Dosage Required            : 100
Population Covered         : 0.1
Vaccine Quantity (In Mill) : 10

Enter 2 to Save and Create another inventory, 1 to Save and go back to Main Menu,
0 to Discard and go back to Main Menu, or Any Number to Redo
```

When the user inputs 1, which is save and go back to main menu, the program will display “Vaccine Details Successfully Saved” and the user will be returned to main menu.



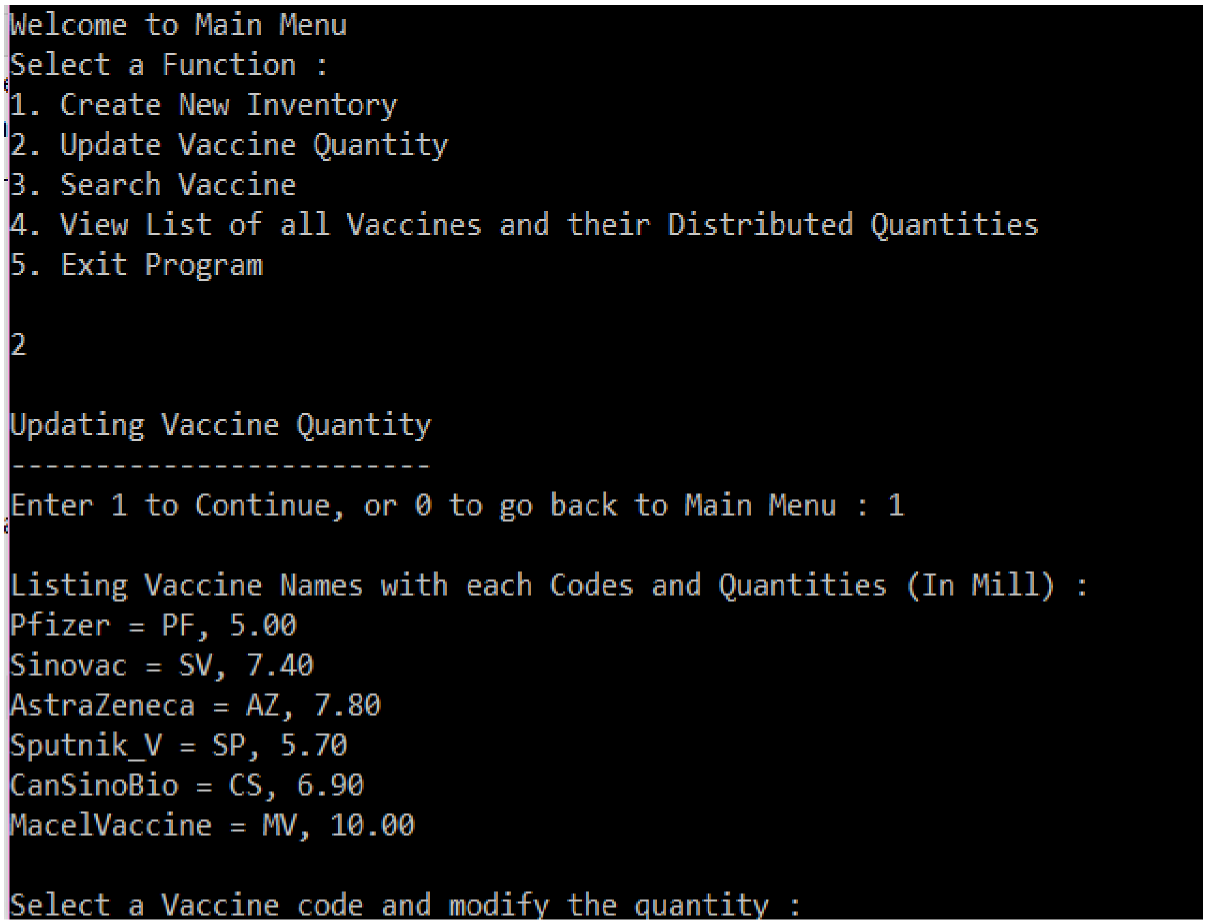
```
Creating New Vaccine Inventory, Enter 1 to Continue, or 0 to go Back : 1
Enter Name of Vaccine      : MacelVaccine
Enter Vaccine Code         : MV
Enter Producing Country    : Indonesia
Dosage Required            : 100
Population Covered         : 0.1
Vaccine Quantity (In Mill) : 10

Enter 2 to Save and Create another inventory, 1 to Save and go back to Main Menu,
0 to Discard and go back to Main Menu, or Any Number to Redo
1

Vaccine Details Successfully Saved

Welcome to Main Menu
Select a Function :
1. Create New Inventory
2. Update Vaccine Quantity
3. Search Vaccine
4. View List of all Vaccines and their Distributed Quantities
5. Exit Program
```

The details from when the user creates the new inventory will be saved in vaccine.txt



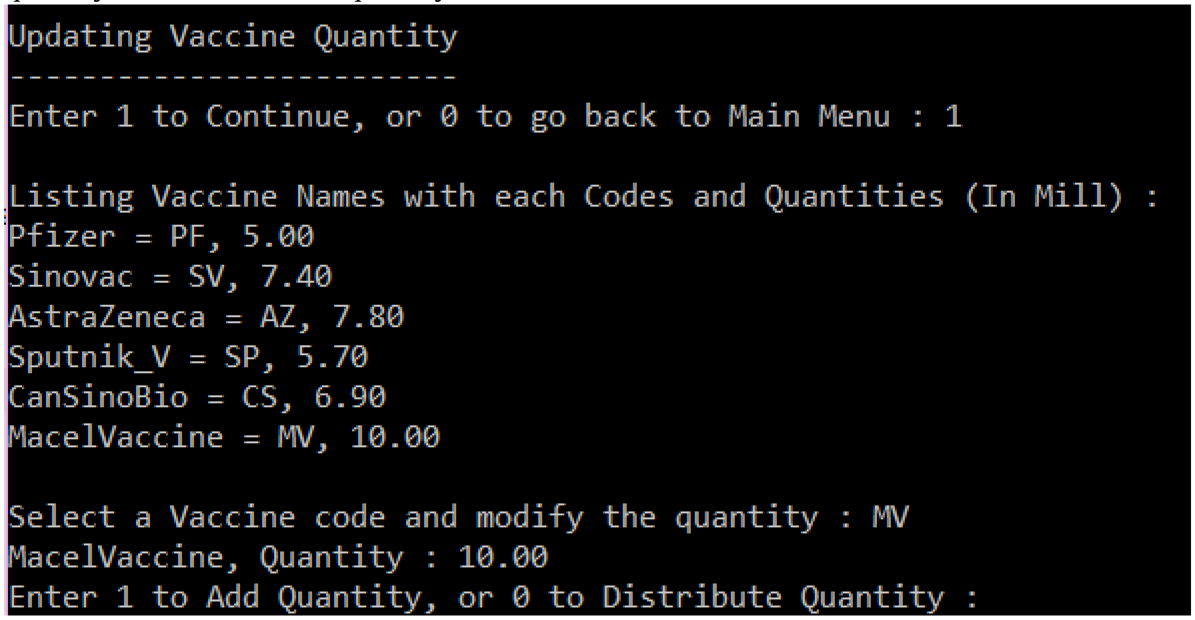
```
Pfizer PF USA 2 50.00 5.00
Sinovac SV China 2 18.80 7.40
AstraZeneca AZ UK 2 10.00 7.80
Sputnik_V SP Russia 2 10.00 5.70
CanSinoBio CS China 1 10.90 6.90
MacelVaccine MV Indonesia 100 0.1 10.0
```

The second function is Update Vaccine Quantity, when the user executes the function, a verify feature will pop up, and again asks the user to continue or go back to main menu. When the user enters 1 to continue, the program will show existing vaccines, vaccine code and their quantities in case the user forgot the vaccine code.



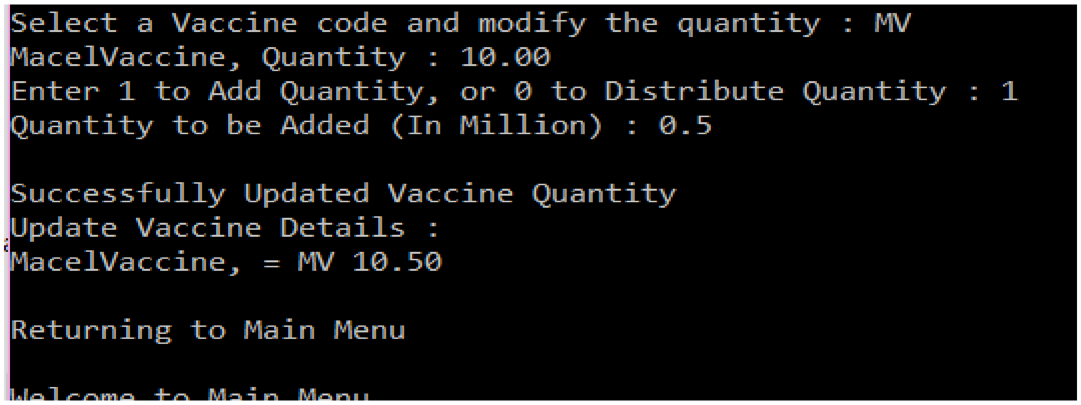
```
Welcome to Main Menu
Select a Function :
1. Create New Inventory
2. Update Vaccine Quantity
3. Search Vaccine
4. View List of all Vaccines and their Distributed Quantities
5. Exit Program

2

Updating Vaccine Quantity
-------------------------
Enter 1 to Continue, or 0 to go back to Main Menu : 1

Listing Vaccine Names with each Codes and Quantities (In Mill) :
Pfizer = PF, 5.00
Sinovac = SV, 7.40
AstraZeneca = AZ, 7.80
Sputnik_V = SP, 5.70
CanSinoBio = CS, 6.90
MacelVaccine = MV, 10.00

Select a Vaccine code and modify the quantity :
```

After the user enters the vaccine code, the program will then ask the user to enter 1 to add quantity, or 0 to distribute quantity,

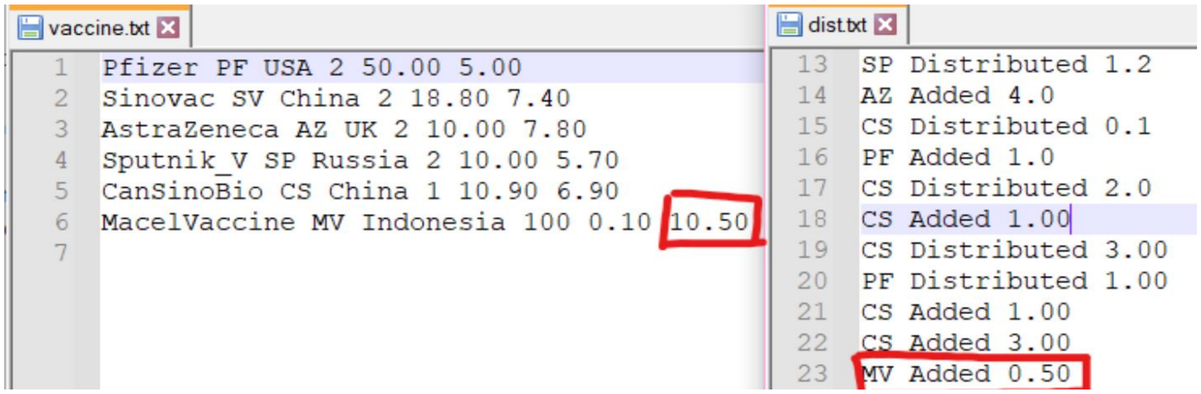


```
Updating Vaccine Quantity
-------------------------
Enter 1 to Continue, or 0 to go back to Main Menu : 1

Listing Vaccine Names with each Codes and Quantities (In Mill) :
Pfizer = PF, 5.00
Sinovac = SV, 7.40
AstraZeneca = AZ, 7.80
Sputnik_V = SP, 5.70
CanSinoBio = CS, 6.90
MacelVaccine = MV, 10.00

Select a Vaccine code and modify the quantity : MV
MacelVaccine, Quantity : 10.00
Enter 1 to Add Quantity, or 0 to Distribute Quantity :
```

After the user selects either to add or distribute quantity, user will be asked the input the quantity, the program will then update the vaccine quantity in vaccine.txt, and will add a history in dist.txt. The user will then be directed to main menu.



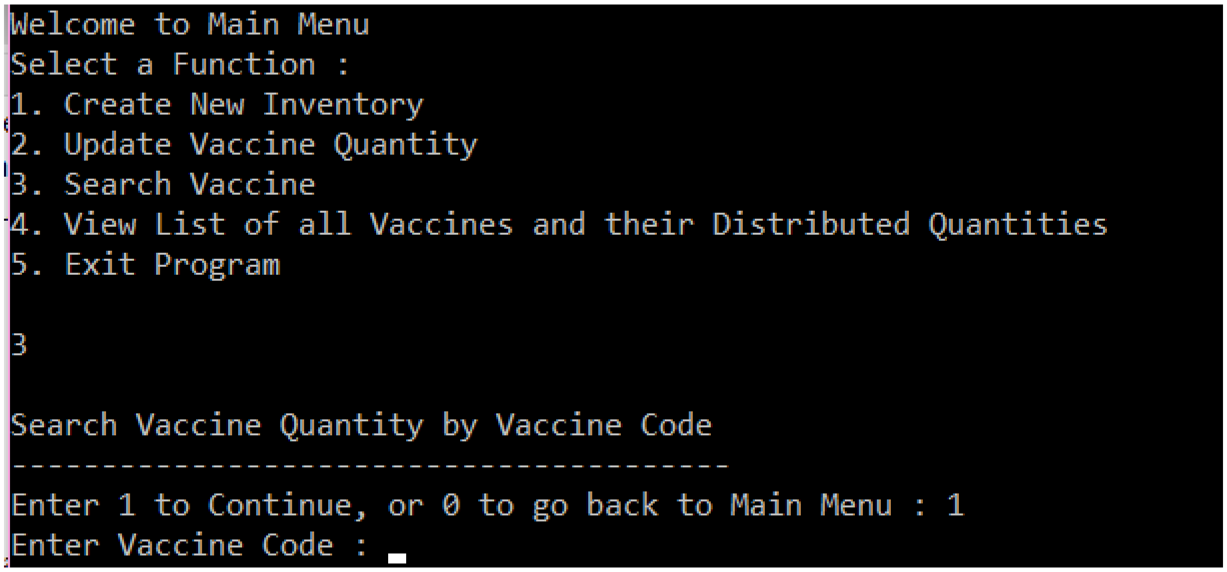
```
Select a Vaccine code and modify the quantity : MV
MacelVaccine, Quantity : 10.00
Enter 1 to Add Quantity, or 0 to Distribute Quantity : 1
Quantity to be Added (In Million) : 0.5

Successfully Updated Vaccine Quantity
Update Vaccine Details :
MacelVaccine, = MV 10.50

Returning to Main Menu
```



```
Pfizer PF USA 2 50.00 5.00
Sinovac SV China 2 18.80 7.40
AstraZeneca AZ UK 2 10.00 7.80
Sputnik_V SP Russia 2 10.00 5.70
CanSinoBio CS China 1 10.90 6.90
MacelVaccine MV Indonesia 100 0.10 10.50
```

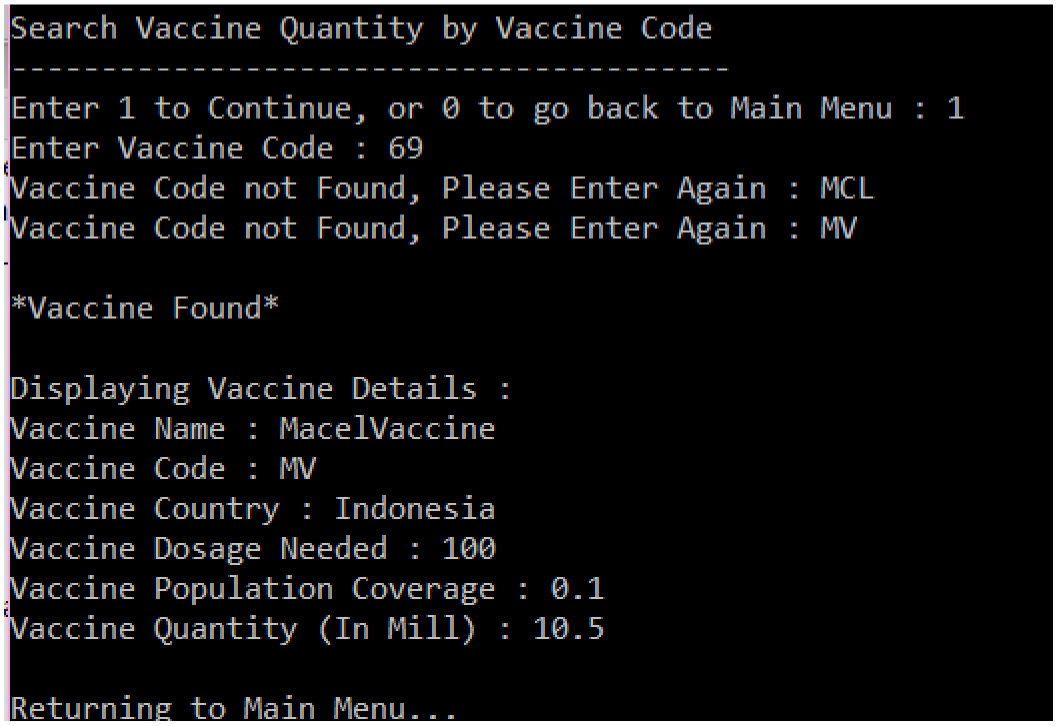
```
SP Distributed 1.2
AZ Added 4.0
CS Distributed 0.1
PF Added 1.0
CS Distributed 2.0
CS Added 1.00
CS Distributed 3.00
PF Distributed 1.00
CS Added 1.00
CS Added 3.00
MV Added 0.50
```

The third function is Search Vaccine Quantity by Vaccine Code, in the main menu, the user needs to input 3 to access this function, then a verify feature will once again pop up, and after verifying, the user will be asked to enter a vaccine code.



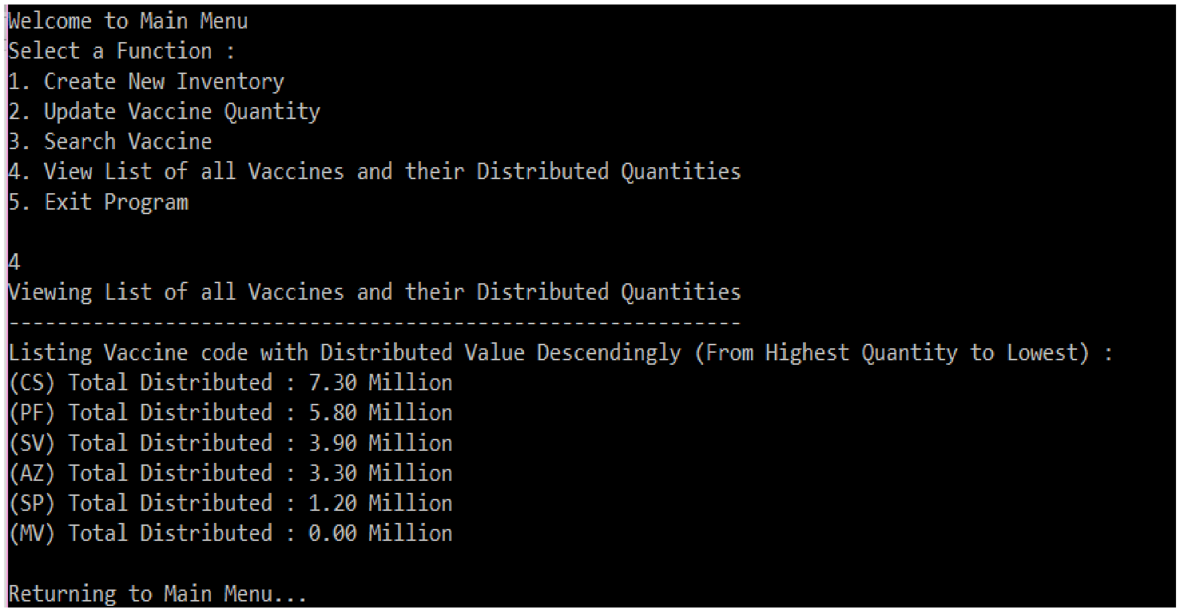
```
Welcome to Main Menu
Select a Function :
1. Create New Inventory
2. Update Vaccine Quantity
3. Search Vaccine
4. View List of all Vaccines and their Distributed Quantities
5. Exit Program

3


Search Vaccine Quantity by Vaccine Code
----------------------------------------
Enter 1 to Continue, or 0 to go back to Main Menu : 1
Enter Vaccine Code : _
```

The program will not work if the user inputs the wrong vaccine code (in the example, 69 and MCL is inputted), once the vaccine code the user input matches the vaccine code in vaccine.txt, the program will display “Vaccine Found”, and the details of the vaccine, the user will then be returned to main menu.



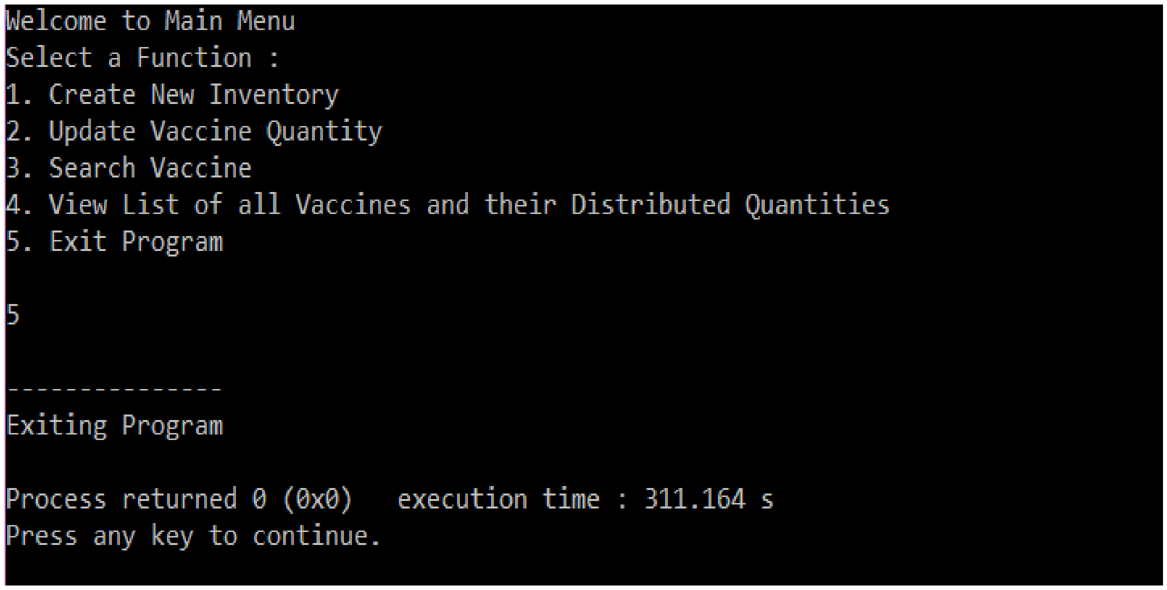
```
Search Vaccine Quantity by Vaccine Code
----------------------------------------
Enter 1 to Continue, or 0 to go back to Main Menu : 1
Enter Vaccine Code : 69
Vaccine Code not Found, Please Enter Again : MCL
Vaccine Code not Found, Please Enter Again : MV

*Vaccine Found*

Displaying Vaccine Details :
Vaccine Name : MacelVaccine
Vaccine Code : MV
Vaccine Country : Indonesia
Vaccine Dosage Needed : 100
Vaccine Population Coverage : 0.1
Vaccine Quantity (In Mill) : 10.5

Returning to Main Menu...
```

The fourth function is to view list of all vaccines and their distributed quantities, this function will sum the total distribution of each vaccine located in dist.txt and display it for the user, this function doesn't have verifying feature since it doesn't take any input from the user and thus doesn't take much time if the user were to select it by accident. After the program lists all the total distribution of the vaccines, the user will then be redirected to the main menu.



```
Welcome to Main Menu
Select a Function :
1. Create New Inventory
2. Update Vaccine Quantity
3. Search Vaccine
4. View List of all Vaccines and their Distributed Quantities
5. Exit Program

4
Viewing List of all Vaccines and their Distributed Quantities
-------------------------------------------------------------
Listing Vaccine code with Distributed Value Descendingly (From Highest Quantity to Lowest) :
(CS) Total Distributed : 7.30 Million
(PF) Total Distributed : 5.80 Million
(SV) Total Distributed : 3.90 Million
(AZ) Total Distributed : 3.30 Million
(SP) Total Distributed : 1.20 Million
(MV) Total Distributed : 0.00 Million

Returning to Main Menu...
```

The last function is called Exit Program, and as it's name suggests, this function will immediately exit the program.



```
Welcome to Main Menu
Select a Function :
1. Create New Inventory
2. Update Vaccine Quantity
3. Search Vaccine
4. View List of all Vaccines and their Distributed Quantities
5. Exit Program

5


---------------
Exiting Program

Process returned 0 (0x0)   execution time : 311.164 s
Press any key to continue.
```

6. Conclusion

It is concluded that the program is designed as the client intended, and have been added extra features to provide user convenience, program flexibility, easy updation, and more.

1. It is concluded the system is able to create a new vaccine inventory, the new vaccine record will be appended to vaccine.txt
2. It is concluded the system is able to update vaccine quantities. The system will have a function to select a specific vaccine and specify either received or distributed quantity. The quantity of the specified vaccine will then be added or subtracted correspondingly in the vaccine.txt file.
   E.g., suppose the initial quantity of vaccine A in vaccine.txt is 3 million. When the company is distributing 1 million vaccines to a hospital, the quantity of vaccine A will be updated to be 2 million instead of 3 million. This distribution will also be recorded into a file called dist.txt.
3. It is concluded the system is able to search a vaccine’s available quantity by using vaccine code, the system is assumed to have a feature where an employee to search for a specific vaccine’s quantity by inputting only the vaccine code.
4. It is concluded the system is able to produce a list of all vaccines’ distributed quantities. The system is assumed to be able to allow the employees to list all distributed vaccines and their accumulated quantities that was recorded in the dist.txt file. These distributed quantities will have to be sorted descendingly (with highest quantity first and lowest quantity last).

7. References

GeeksforGeeks, 2020. *C Program to Swap two Numbers.* [Online]
Available at: https://www.geeksforgeeks.org/c-program-swap-two-numbers/
[Accessed 25 May 2021].

Jabberwocky, 2020. *Counting the number of lines in a .txt file in c.* [Online]
Available at: https://stackoverflow.com/questions/62059123/counting-the-number-of-lines-in-a-txt-file-in-c
[Accessed 29 5 2021].