Company name

Description automatically generated

**INDIVIDUAL ASSIGNMENT**

**TECHNOLOGY PARK MALAYSIA**

**CT018-3-1-ICP**

**INTRODUCTION TO C PROGRAMMING**

**APU-APD1F2111/CE/CGD/CS/CYB/CS(DF)/CS(IS)/EEE/IT/ME/PE/TE**

**NAME: SHAIK YASEEN**

**ID:**  **TP062920**

**HAND IN DATE: 14th AUG 2022**

**WEIGHTAGE: 50%**

**INSTRUCTIONS TO CANDIDATES:**

1. Submit your assignment online in Moodle unless advised otherwise
2. Late submission will be awarded zero(0) unless Extenuating Circumstances (EC) are upheld
3. Cases of plagiarism will be penalized
4. You must obtain at least 50% in each component to pass this module

**TABLE OF CONTENT:**

**Introduction …………………...................................................................................................... 3**

**Assumptions ..................................................................................................................................3**

**Design of the program...................................................................................................................4**

**Pseudocode .................................................................................................................................... 4**

**Flowcharts .................................................................................................................................. 12**

**Additional Features ................................................................................................................... 20**

**Sample Outputs and Inputs ...................................................................................................... 21**

**Conclusion ...................................................................................................................................25**

**References ................................................................................................................................... 26**

**INTRODUCTION:**

The coronavirus, often known as SARS-CoV-2, is a lethal virus that has caused an epidemic that has spread globally (severe acute respiratory syndrome coronavirus 2). The World Health Organization refers to the condition caused by a coronavirus as "coronavirus disease" (COVID-19) There are no well-established treatments or specific medications that can be utilised to treat patients. In Malaysia, the vaccination campaign has just begun. The company has tasked us with creating a covid -19 Donations management system to aid in the organisation of the process.

**ASSUMPTIONS:**

I believe the following abilities I can contribute to the Donations Management System will be helpful in these trying times.

• Making an inventory.

• Refresh contribution totals.

• Use the contribution code to search for donations and their available quantities.

• Create a list of all donations, together with the amounts that were disbursed.

By implementing a program with these features, the corporation will just be capable of keeping track of what is in the stock, however many fresh stocks they are receiving, and what immunizations and mask supplies they are supplying.

**DESIGN OF THE PROGRAM:**

**PSEUDOCODE:**

BEGIN

print"\n=============Malaysia Red Crescent Society ==============\n”

print"\n========= COVID-19 DONATION MANAGEMENT SYSTEM ===========\n”

int choice

do while (1)

print"\n1. Inventory Creation \n”

print"2. Update Donation Quantities \n”

print"3. Search Donations by Donation Code \n”

print"4. List Of All Donations and Distributed Quantities \n”

print"5. Exit \n”

print"\nEnter your choice : “

Declare("%d", &choice)

switch (choice)

case 1:

// Inventory Creation

inventory creation()

case 2:

// Update Donation Quantities

update donation()

case 3:

// Search Donations by Donation Code

search()

case 4:

// List Of All Donations and Their Distributed Quantities

bubble sort()

case 5:

print"\*\*\*\*\*\*\*-- Have A Nice Day! System Shutdown --\*\*\*\*\*\*\*\n”

exit(0) // terminates the complete program execution

default:

print"\n\n\*\*\*\*-- Enter Correct Input! --\*\*\*\*\n\n”

print"\n\n\t\t\*\*\*-- Have A Nice Day! --\*\*\*\n\n\n”

END CASE

END DO WHILE

END FUNCTION

Function inventory creation()

IF (check\_donation\_textfile\_exist())

print"\n\*\*\*-- Donation Text File Already Exist! --\*\*\*\n”

read inventory()

print inventory()

return

else

FILE \*fp

fp = fopen("donation.txt", "w”

IF (fp == NULL)

print"\n\*\*\*-- Error in Opening File! --\*\*\*\n”

return

else

initialize inventory()

print inventory()

print"\n\*\*\*-- Donation Text File Created Successfully! --\*\*\*\n”

fclose(fp)

END IF

END FUNCTION

Function bool check\_donation\_textfile\_exist()

FILE \*fp

fp = fopen("donation.txt", "r”

IF (fp == NULL)

return false

else

return true

END FUNCTION

Function initialize\_inventory()

// IF donation.txt does not exist then create a new text file and store the data in the struct array

FILE \*fp

fp = fopen("donation.txt", "w”

IF (fp == NULL)

print"\n\*\*\*-- Error in Opening File! --\*\*\*\n”

return

else

strcpy(inventory list[0].name\_of\_supply, "ContactlessThermometer”

strcpy(inventory\_list[0].supply\_code, "CT”

strcpy(inventory\_list[0].donator\_country, "Japan”

inventory\_list[0].no\_of\_shipments = 1

inventory\_list[0].quantity\_received = 1.2

strcpy(inventory\_list[1].name\_of\_supply, "HandSanitizer”

strcpy(inventory\_list[1].supply\_code, "HS”

strcpy(inventory\_list[1].donator\_country, "USA”

inventory\_list[1].no\_of\_shipments = 1

inventory\_list[1].quantity\_received = 3.5

strcpy(inventory\_list[2].name\_of\_supply, "FaceMask”

strcpy(inventory\_list[2].supply\_code, "FM”

strcpy(inventory\_list[2].donator\_country, "China”

inventory\_list[2].no\_of\_shipments = 2

inventory\_list[2].quantity\_received = 120.0

strcpy(inventory\_list[3].name\_of\_supply, "SurgicalMask”

strcpy(inventory\_list[3].supply\_code, "SM”

strcpy(inventory\_list[3].donator\_country, "China”

inventory\_list[3].no\_of\_shipments = 2

inventory\_list[3].quantity\_received = 38.0

strcpy(inventory\_list[4].name\_of\_supply, "OxygenMask”

strcpy(inventory\_list[4].supply\_code, "OM”

strcpy(inventory\_list[4].donator\_country, "SaudiArabia”

inventory\_list[4].no\_of\_shipments = 2

inventory\_list[4].quantity\_received = 9.0

// write the struct array to the text file

FOR i FROM 0 TO < 5 i

fprintfp, "%s\t%s\t%s\t%d\t%f\n", inventory\_list[i].name\_of\_supply, inventory\_list[i].supply\_code,

inventory\_list[i].donator\_country, inventory\_list[i].no\_of\_shipments,

inventory\_list[i].quantity\_received)

fclose(fp)

END FOR

END IF

END FUNCTION

Function read\_inventory()

// IF donation.txt exits then read donation.txt file and store the data in the struct array

IF (check\_donation\_textfile\_exist())

FILE \*fp

fp = fopen("donation.txt", "r”

IF (fp == NULL)

print"\n\t\t\t\033[031mError in Opening File!\033[0m\n\n”

return

else

// read the text file and store the data in the struct array

FILE \*fp

fp = fopen("donation.txt", "r”

FOR i FROM 0 TO < 5

fDeclare(fp, "%s", inventory\_list[i].name\_of\_supply)

fDeclare(fp, "%s", inventory\_list[i].supply\_code)

fDeclare(fp, "%s", inventory\_list[i].donator\_country)

fDeclare(fp, "%d", &inventory\_list[i].no\_of\_shipments)

fDeclare(fp, "%f", &inventory\_list[i].quantity\_received)

fclose(fp)

END IF

END FOR

END FUNCTION

Function print\_inventory()

print"\n||Name of Supply|| ||Supply Code|| ||Donator|| ||No. of Shipment|| ||Quantity Received (millions)||

print"%s%10s%17s%16d%26.3f\n", inventory\_list[0].name\_of\_supply, inventory\_list[0].supply\_code,

inventory\_list[0].donator\_country, inventory\_list[0].no\_of\_shipments, inventory\_list[0].quantity\_received)

print"%s%19s%16s%17d%26.3f\n", inventory\_list[1].name\_of\_supply, inventory\_list[1].supply\_code,

inventory\_list[1].donator country, inventory list[1].no\_of\_shipments, inventory\_list[1].quantity\_received)

print"%s%24s%17s%16d%26.3f\n", inventory\_list[2].name\_of\_supply, inventory\_list[2].supply\_code,

inventory\_list[2].donator\_country, inventory\_list[2].no\_of\_shipments, inventory\_list[2].quantity\_received)

print"%s%20s%17s%16d%26.3f\n", inventory\_list[3].name\_of\_supply, inventory\_list[3].supply\_code,

inventory\_list[3].donator\_country, inventory\_list[3].no\_of\_shipments, inventory\_list[3].quantity\_received)

print"%s%22s%20s%13d%26.3f\n", inventory\_list[4].name\_of\_supply, inventory\_list[4].supply\_code,

inventory\_list[4].donator\_country, inventory\_list[4].no\_of\_shipments, inventory\_list[4].quantity\_received)

END FUNCTION

Function update\_donation()

read\_inventory()

char search\_code[3]

float quantity

print"Enter the supply code: “

Declare("%s", search\_code)

FOR i FROM 0 TO < 5

IF (strcmp(inventory\_list[i].supply\_code, search\_code) == 0)

print"Name of Supply: %s\nSupply Code: %s\nDonator: %s\nNo. of shipment:%d\nQuantity Received (millions): %.3f\n",

inventory\_list[i].name\_of\_supply, inventory\_list[i].supply\_code,

inventory\_list[i].donator\_country, inventory\_list[i].no\_of\_shipments,

inventory\_list[i].quantity\_received)

print"Do you want to donate supplies to other countries or receive donations from other countries?\n”

print"1. Donate\n”

print"2. Receive\n”

int choice

Declare("%d", &choice)

IF (choice == 1)

print"Enter the quantity to donate: “

Declare("%f", &quantity)

// check IF the quantity is greater than the quantity received

IF(quantity > inventory\_list[i].quantity\_received)

print"\n\*\*\*-- Error: Stock Unavailable for that amount! --\*\*\*\n”

return

inventory\_list[i].quantity\_received -= quantity

write\_dist.quantity = quantity

strcpy(write\_dist.supply\_code, inventory\_list[i].supply\_code)

print"\n\*\*\*-- Donation Successful! --\*\*\*\n”

else IF (choice == 2)

print"Enter the quantity to receive: “

Declare("%f", &quantity)

inventory\_list[i].quantity\_received += quantity

print"\n\*\*\*-- Received Successful! --\*\*\*\n”

else

print"\*\*\*-- Invalid choice! --\*\*\*\n”

write\_inventory()

write\_distribution()

END IF

END FUNCTION

Function write\_inventory()

FILE \*f

f = fopen("donation.txt", "w”

FOR i FROM 0 TO < 5

Print file, "%s %s %s %d %f\n", inventory\_list[i].name\_of\_supply, inventory\_list[i].supply\_code,

inventory\_list[i].donator\_country, inventory\_list[i].no\_of\_shipments,

inventory\_list[i].quantity\_received)

fclose(f)

END FOR

END FUNCTION

Function write\_distribution()

FILE \*f

f = fopen("dist.txt", "a”

print "%s %f\n", write\_dist.supply\_code, write\_dist.quantity”

fclose(f)

END FUNCTION

Function search()

read\_inventory()

char search\_code[3]

print"Enter the supply code to search: “

Declare("%s", search\_code)

FOR i FROM 0 TO < 5

IF (strcmp(inventory\_list[i].supply\_code, search\_code) == 0)

print"Name of Supply: %s\nSupply Code: %s\nDonator: %s\nNo. of shipment:%d\nQuantity Received (millions): %.3f\n",

inventory\_list[i].name\_of\_supply, inventory\_list[i].supply\_code,

inventory\_list[i].donator\_country, inventory\_list[i].no\_of\_shipments,

inventory\_list[i].quantity\_received)

END FOR

END FUNCTION

Fuction bubble\_sort()

int ch = 0

int lines = 0

FILE \*fp

fp = fopen("dist.txt", "r”

IF (fp == NULL)

print"\n\*\*\*Error in Opening File!\*\*\*\n”

return

while (!feof(fp))

ch = fgetc(fp)

IF (ch == '\n')

lines++

struct distribution\_list dist[lines]

// read the text file and store the data in the struct array

FILE \*f

f = fopen("dist.txt", "r”

FOR i FROM 0 TO < LINE

Declare(f, "%s", dist[i].supply\_code)

Declare(f, "%f", &dist[i].quantity)

fclose(f)

// print the unsorted array

for (int i = 0 i < lines i++)

print"%s %.3f\n", dist[i].supply\_code, dist[i].quantity”

// bubble sort in descending order

FOR i FROM 0 TO < LINE

FOR J FROM 0 TO < LINE-1

IF (dist[j].quantity < dist[j + 1].quantity)

struct distribution\_list temp

temp = dist[j]

dist[j] = dist[j + 1]

dist[j + 1] = temp

// print the sorted data

FOR i FROM 0 TO < LINE

print"%s %.3f\n", dist[i].supply\_code, dist[i].quantity”

fclose(fp)

END IF

END WHILE

END FOR

END FOR

END FOR

END FUNCTION

END

**FLOWCHART:**

1. Source. c
2. Inventory Creation.c
3. Initialize inventory.c
4. Printing inventory.c
5. Update donation.c
6. Write inventory.c
7. Write donation.c
8. Read inventory.c
9. Bubble sort. h

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

A picture containing diagram

Description automatically generated

Diagram, schematic

Description automatically generated

**ADDITIONAL FEATURES:**

Text

Description automatically generated

Text

Description automatically generated

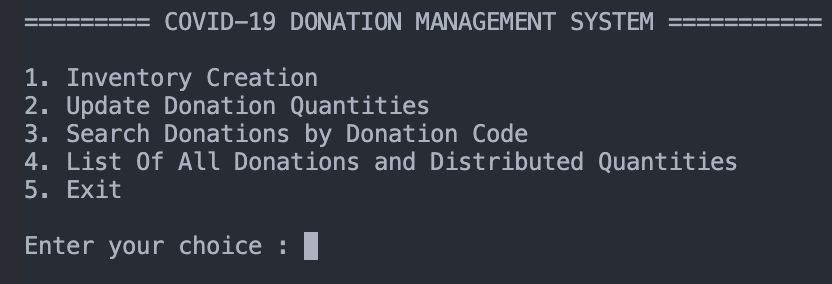
As you can see, the program will continue to function without pausing even when the user enters wrong information and displays an error message.

Text

Description automatically generated

when the user enters the option the program will continue to function without pausing and the menu bar will appear.

**SAMPLE OUTPUTS AND INPUTS:**



This is my program's menu. The user can input a number between 1 and 5 here to see the

contents of the submenus.

A picture containing graphical user interface

Description automatically generated

The donation record that already exists in the donation text. file is displayed if the user inputs 1, and we can view the name of the supplies, supply codes, donors, the number of shipments, and ultimately the quantity delivered in millions.

Graphical user interface, text, application

Description automatically generated

In the event that the user inputs 2, the software displays Update Donation Quantities, prompts the user to enter the supply code, and then displays the name of the supply, the supply's code, the donor, the number of shipments, and the amount received in millions.

Text

Description automatically generated

The programme will ask the user how much quantity they wish to contribute if they input donate. The application will then deduct the quantity from the existing value.

Text

Description automatically generated

The programme prompts the user to specify how much the amount they wish to receive if they enter Receive. The software will then subtract the quantity from current value.

Text

Description automatically generated

If the user types in 3, it displays information on donations by donation code. For instance, if the user types in CT, the software displays the name of the supply: Contactless thermometer, code : CT, Donator: Japan, Shipment Number: 1, and Amount Received (Millions): 780.200

Text

Description automatically generated

When the user inputs 4, all donation and associated distributed quantities are displayed. The scattered quantities are arranged from highest to lowest in increasing order.

Text

Description automatically generated

The User can escape by inputting 6 if he so desires.

**CONCLUSION:**

I'm pleased to have completed introducing the program's outstanding features in adding to its basic capabilities. I'm hopeful that this initiative will help make the government's and business's immunization process easier. I've programmed the software with all of my assumptions. Building software with these traits was difficult at first, but after I worked it out, it was simple and efficient. I ended up mastering the C programming language also was happy with the results of my work on this program.

**REFERENCES:**

*Best C programming books for beginners & expert [2022]*. InterviewBit. (2022, January 13). Retrieved August 13, 2022, from https://www.interviewbit.com/blog/c-programming-books/

Raman. (2022, March 15). *C programming books for beginners and advanced*. GreatLearning Blog: Free Resources what Matters to shape your Career! Retrieved August 14, 2022, from https://www.mygreatlearning.com/blog/c-programming-books/