#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <arpa/inet.h>

#define PORT 8080

#define BUFFER\_SIZE 1024

int main() {

int server\_fd, new\_socket;

struct sockaddr\_in address;

int opt = 1;

int addrlen = sizeof(address);

char buffer[BUFFER\_SIZE] = {0};

const char \*response = "Hello from the Grow microservice!";

if ((server\_fd = socket(AF\_INET, SOCK\_STREAM, 0)) == 0) {

perror("Socket failed");

exit(EXIT\_FAILURE);

}

if (setsockopt(server\_fd, SOL\_SOCKET, SO\_REUSEADDR, &opt, sizeof(opt))) {

perror("setsockopt");

exit(EXIT\_FAILURE);

}

address.sin\_family = AF\_INET;

address.sin\_addr.s\_addr = INADDR\_ANY;

address.sin\_port = htons(PORT);

if (bind(server\_fd, (struct sockaddr \*)&address, sizeof(address)) < 0) {

perror("Bind failed");

exit(EXIT\_FAILURE);

}

if (listen(server\_fd, 3) < 0) {

perror("Listen failed");

exit(EXIT\_FAILURE);

}

printf("Server is listening on port %d\n", PORT);

while ((new\_socket = accept(server\_fd, (struct sockaddr \*)&address, (socklen\_t\*)&addrlen)) >= 0) {

read(new\_socket, buffer, BUFFER\_SIZE);

printf("Received: %s\n", buffer);

send(new\_socket, response, strlen(response), 0);

close(new\_socket);

}

return 0;

}

MCQ:

1. C) To create a new stock
2. D) Queue
3. B) To update the price of a stock

Programming Questions:

1)

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_STOCKS 100

typedef struct {

char name[50];

float price;

int quantity;

} Stock;

Stock stocks[MAX\_STOCKS];

int stockCount = 0;

void addStock(const char \*name, float price) {

strcpy(stocks[stockCount].name, name);

stocks[stockCount].price = price;

stocks[stockCount].quantity = 0;

stockCount++;

}

void displayStocks() {

printf("\nCurrent Stocks:\n");

for (int i = 0; i < stockCount; i++) {

printf("Stock: %s | Price: %.2f | Quantity: %d\n", stocks[i].name, stocks[i].price, stocks[i].quantity);

}

}

void buyStock(const char \*name, int quantity) {

for (int i = 0; i < stockCount; i++) {

if (strcmp(stocks[i].name, name) == 0) {

stocks[i].quantity += quantity;

printf("Bought %d shares of %s at %.2f each.\n", quantity, stocks[i].name, stocks[i].price);

return;

}

}

printf("Stock %s not found.\n", name);

}

void sellStock(const char \*name, int quantity) {

for (int i = 0; i < stockCount; i++) {

if (strcmp(stocks[i].name, name) == 0) {

if (stocks[i].quantity >= quantity) {

stocks[i].quantity -= quantity;

printf("Sold %d shares of %s at %.2f each.\n", quantity, stocks[i].name, stocks[i].price);

} else {

printf("Not enough shares of %s to sell.\n", name);

}

return;

}

}

printf("Stock %s not found.\n", name);

}

int main() {

// Adding some stocks to the simulator

addStock("AAPL", 150.00);

addStock("GOOGL", 2800.00);

addStock("AMZN", 3400.00);

int choice;

char stockName[50];

int quantity;

while (1) {

printf("\nStock Market Simulator\n");

printf("1. Display Stocks\n");

printf("2. Buy Stock\n");

printf("3. Sell Stock\n");

printf("4. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

displayStocks();

break;

case 2:

printf("Enter stock name to buy: ");

scanf("%s", stockName);

printf("Enter quantity to buy: ");

scanf("%d", &quantity);

buyStock(stockName, quantity);

break;

case 3:

printf("Enter stock name to sell: ");

scanf("%s", stockName);

printf("Enter quantity to sell: ");

scanf("%d", &quantity);

sellStock(stockName, quantity);

break;

case 4:

printf("Exiting the simulator.\n");

exit(0);

default:

printf("Invalid choice. Please try again.\n");

}

}

return 0;

}

2)

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_STOCKS 100

typedef struct {

char name[50];

float price;

int quantity;

} Stock;

Stock stocks[MAX\_STOCKS];

int stockCount = 0;

void addStock(const char \*name, float price) {

strcpy(stocks[stockCount].name, name);

stocks[stockCount].price = price;

stocks[stockCount].quantity = 0;

stockCount++;

}

void displayStocks() {

printf("\nCurrent Stocks:\n");

for (int i = 0; i < stockCount; i++) {

printf("Stock: %s | Price: %.2f | Quantity: %d\n", stocks[i].name, stocks[i].price, stocks[i].quantity);

}

}

void buyStock(const char \*name, int quantity) {

for (int i = 0; i < stockCount; i++) {

if (strcmp(stocks[i].name, name) == 0) {

stocks[i].quantity += quantity;

printf("Bought %d shares of %s at %.2f each.\n", quantity, stocks[i].name, stocks[i].price);

return;

}

}

printf("Stock %s not found.\n", name);

}

void sellStock(const char \*name, int quantity) {

for (int i = 0; i < stockCount; i++) {

if (strcmp(stocks[i].name, name) == 0) {

if (stocks[i].quantity >= quantity) {

stocks[i].quantity -= quantity;

printf("Sold %d shares of %s at %.2f each.\n", quantity, stocks[i].name, stocks[i].price);

} else {

printf("Not enough shares of %s to sell.\n", name);

}

return;

}

}

printf("Stock %s not found.\n", name);

}

void search\_stock(const char \*name) {

for (int i = 0; i < stockCount; i++) {

if (strcmp(stocks[i].name, name) == 0) {

printf("Stock: %s | Price: %.2f | Quantity: %d\n", stocks[i].name, stocks[i].price, stocks[i].quantity);

return;

}

}

printf("Stock %s not found.\n", name);

}

int main() {

addStock("AAPL", 150.00);

addStock("GOOGL", 2800.00);

addStock("AMZN", 3400.00);

int choice;

char stockName[50];

int quantity;

while (1) {

printf("\nStock Market Simulator\n");

printf("1. Display Stocks\n");

printf("2. Buy Stock\n");

printf("3. Sell Stock\n");

printf("4. Search Stock\n");

printf("5. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

displayStocks();

break;

case 2:

printf("Enter stock name to buy: ");

scanf("%s", stockName);

printf("Enter quantity to buy: ");

scanf("%d", &quantity);

buyStock(stockName, quantity);

break;

case 3:

printf("Enter stock name to sell: ");

scanf("%s", stockName);

printf("Enter quantity to sell: ");

scanf("%d", &quantity);

sellStock(stockName, quantity);

break;

case 4:

printf("Enter stock name to search: ");

scanf("%s", stockName);

search\_stock(stockName);

break;

case 5:

printf("Exiting the simulator.\n");

exit(0);

default:

printf("Invalid choice. Please try again.\n");

}

}

return 0;

}

3)

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct Stock {

char name[50];

float price;

int quantity;

struct Stock\* next;

} Stock;

Stock\* head = NULL;

Stock\* createStock(const char\* name, float price, int quantity) {

Stock\* newStock = (Stock\*)malloc(sizeof(Stock));

strcpy(newStock->name, name);

newStock->price = price;

newStock->quantity = quantity;

newStock->next = NULL;

return newStock;

}

void addStock(const char\* name, float price, int quantity) {

Stock\* newStock = createStock(name, price, quantity);

newStock->next = head;

head = newStock;

printf("Stock %s added successfully.\n", name);

}

void deleteStock(const char\* name) {

Stock\* current = head;

Stock\* previous = NULL;

while (current != NULL && strcmp(current->name, name) != 0) {

previous = current;

current = current->next;

}

if (current == NULL) {

printf("Stock %s not found.\n", name);

return;

}

if (previous == NULL) {

head = current->next; // Deleting the head

} else {

previous->next = current->next; // Bypass the current node

}

free(current);

printf("Stock %s deleted successfully.\n", name);

}

void displayStocks() {

Stock\* current = head;

if (current == NULL) {

printf("No stocks available.\n");

return;

}

printf("\nCurrent Stocks:\n");

while (current != NULL) {

printf("Stock: %s | Price: %.2f | Quantity: %d\n", current->name, current->price, current->quantity);

current = current->next;

}

}

int main() {

int choice;

char stockName[50];

float stockPrice;

int stockQuantity;

while (1) {

printf("\nStock Management System\n");

printf("1. Add Stock\n");

printf("2. Delete Stock\n");

printf("3. Display Stocks\n");

printf("4. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

printf("Enter stock name: ");

scanf("%s", stockName);

printf("Enter stock price: ");

scanf("%f", &stockPrice);

printf("Enter stock quantity: ");

scanf("%d", &stockQuantity);

addStock(stockName, stockPrice, stockQuantity);

break;

case 2:

printf("Enter stock name to delete: ");

scanf("%s", stockName);

deleteStock(stockName);

break;

case 3:

displayStocks();

break;

case 4:

printf("Exiting the program.\n");

exit(0);

default:

printf("Invalid choice. Please try again.\n");

}

}

return 0;

}