**개발 완료 보고서**

제출일 : 2023년 12월 25일

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **참여자** | | 이수빈 | | | |
| **개발프로젝트 소개** | | | | | |
| **활동일시** | | 2023.12.24 - 2023.12.25 | | **장소** | 집 |
| **주요 주제** | | 1. 크리스마스 케이크 예약 주문 프로그램을 만든다. 2. 서버와 클라이언트 관계를 이용한다. | | | |
| **개발목적** | | 크리스마스에 즐겁게 코딩하기 | | | |
| **개발환경** | | ﻿ Windows/C/visual studio 2022 | | | |
| 개인별  업무분담 | 이수빈 | 메인 ,로직, UI | | | |
| **요구 분석서** | |  | | | |
| **일정표** | | 12/23 - 개발계획서, (간단한)요구분석서, 순서도 작성  12/24 – 불꽃 코딩!! 할 수 있다!!  12/25 – 디버그 및 코드 점검. 개발 완료 보고서 작성 및 제출(13:00까지) | | | |
| **﻿**  **코드**  **설명**  **﻿** | 서버 | | typedef struct  {  char menu[50];  int price;  }Menu; //메뉴판  - 메뉴 정보를 저장하는 구조체를 선언하였다  //주문정보 저장할 파일 열기  char line[BUF\_SIZE] = { 0 };  FILE\* fp = fopen("orderList.txt", "a+");  while (1) //주문 번호 누적시키기  {  fgets(line, BUF\_SIZE, fp);  if (feof(fp)) break;  if (strstr(line, "주문 번호") != NULL) ordernum++;  }  - 서버 시작 시 기존에 있는(없을 시 생성) 데이터 파일을 읽어 주문 번호를 누적시킨다.  //메뉴 정보 전송해야 함  for ( i = 0;i < MENU\_NUM;i++)  {  //길이를 같이 보내려 하지 말고 길이 먼저 전송하고 그만큼 읽어들이게!  sprintf(SndMsg, "%zd", strlen(cake[i].menu));  send(hClntSock, SndMsg, strlen(SndMsg), 0); //길이 전송  recv(hClntSock, buf, BUF\_SIZE, 0); //확인메시지 수신  send(hClntSock, cake[i].menu, strlen(cake[i].menu), 0); //메뉴 이름 전송  recv(hClntSock, buf, BUF\_SIZE, 0); //확인메시지 수신  sprintf(SndMsg, "%d", cake[i].price); //가격은 아스키 코드값을 넘어가니 문자열에 넣어서  send(hClntSock, SndMsg, strlen(SndMsg), 0);  recv(hClntSock, buf, BUF\_SIZE, 0); //확인메시지 수신  } memset(buf, 0, BUF\_SIZE); memset(SndMsg, 0, BUF\_SIZE);  - 서버에 저장된 메뉴 정보를 담은 구조체 배열의 내용을 클라이언트에 전송한다.  - 정수형은 그대로 전송할 수 도 있지만 문자열에 담아 전송하는 연습을 해보았다  //주문자 정보 저장  if (recv(hClntSock, buf, BUF\_SIZE, 0) > 0) //주문자 이름 길이 수신  - 주문 정보 전송 후 서버는 이 조건문에서 대기상태에 빠진다  - 클라이언트가 비정상으로 접속 종료 할 시 해당 조건문을 통과하지 못하고 새로운 클라이언트의 연결을 대기하도록 한다.  send(hClntSock, (char\*)&ordernum, sizeof(int), 0);//주문 번호 전송  strLen = atoi(buf); recLen = 0; fputs("주문자 이름: ",fp);fputs("주문자 이름: ", stdout);  while (strLen != recLen) //주문자 이름 수신  {  recv(hClntSock, buf, 1, 0); recLen++;  fputc(buf[0], fp); //한글자씩 입력  fputc(buf[0], stdout); //화면에도 출력  }  - 주문자의 이름이 서버로 수신되어 데이터 파일에 저장되고 화면에 출력된다.  recv(hClntSock, buf, BUF\_SIZE, 0); //전화번호 길이 수신  send(hClntSock, "fin", 4, 0); //수신 확인 메시지  strLen = atoi(buf); recLen = 0; fputs("주문자 전화번호: ", fp);fputs("주문자 전화번호: ", stdout);  while (strLen != recLen) //주문자 전화번호 수신  {  recv(hClntSock, buf, 1, 0); recLen++;  fputc(buf[0], fp); //한글자씩 입력  fputc(buf[0], stdout); //화면에도 출력  }  - 주문자의 전화번호가 서버로 수신되어 데이터 파일에 저장되고 화면에 출력된다.  recv(hClntSock, (char\*)&listNum, sizeof(int), 0); //장바구니 리스트 개수 수신  send(hClntSock, "fin", 4, 0); //수신 확인 메시지  fputs("주문 내역>\n", fp);fputs("주문 내역>\n", stdout);  for ( i = 0;i < listNum;i++) //주문내역 수신  {  recv(hClntSock, (char\*)&menuIdx, sizeof(int), 0); //메뉴 인덱스 수신  fprintf(fp, "%d) %17s : %6d원\n", i + 1, cake[menuIdx].menu, cake[menuIdx].price);  printf("%d) %17s : %6d원\n", i + 1, cake[menuIdx].menu, cake[menuIdx].price);  totalPrice += cake[menuIdx].price;  send(hClntSock, "fin", 4, 0); //수신 확인 메시지  }  fprintf(fp, "\t 총 가격 : % 6d원 \n", totalPrice);  - 먼저 장바구니에 담긴 케이크의 개수를 읽어온다.  - 읽어온 개수 만큼 메뉴의 인덱스를 읽어와 해당 메뉴의 이름과 가격 정보가 데이터 파일에 저장되고 화면에 출력된다.  - 서버 측 총 가격을 저장하는 변수에 누적해서 합해지고 메뉴 인덱스를 모두 읽어온 후 총 합을 데이터 파일에 함께 저장한다. | | |
| 클라이언트 | | typedef struct  {  int count; //메뉴 개수  int menu[10]; //메뉴 인덱스 저장 (이름->인덱스로 변경)  int price[10]; //메뉴가격 저장  int Tprice; //총 가격 저장  char name[20]; //회원 이름 저장  char phone[20]; //핸드폰 번호 저장  char pay[10]; //결제 방법 저장  }Slist; //장바구니  - 장바구니에 담길 케이크의 개수와 그 목록, 주문자 정보를 저장할 구조체를 선언한다.  void PrintMenu(Menu\* menu);  - 메뉴판을 화면에 출력하는 함수.  - 메뉴 구조체 배열의 내용을 고객에게 보여준다.  void PrintList(Slist list,Menu\* menu);  - 장바구니를 화면에 출력하는 함수.  - 구매목록과 총액을 고객에게 보여준다.  void AddMenu(Menu\* menu, Slist\* list,int num);  - 장바구니에 고객이 선택한 케이크를 추가하는 함수.  - menu구조체 배열의 인덱스가 num인 요소의 정보를 장바구니 list에 추가한다.  void Modifylist(Slist\* list,int num);  - 장바구니에 담긴 케이크를 제거해주는 함수.  - 고객이 선택 취소한 케이크를 list에서 제거해준다.  int Pay(Slist\* list,Menu\* menu);  - 결제를 진행하는 함수  - 장바구니 목록을 다시 보여주며 결제를 계속할 것인지 묻는다  🡪 추가 구매 혹은 선택 취소를 원할 시 주문화면으로 돌아갈 수 있다.  - 주문자의 이름과 전화번호를 입력 받아 화면에 출력하고 정보가 맞는지 확인한다.  - 결제 방법을 선택하여 선결제를 진행한다.  void SndList(SOCKET\* sock,Slist list);  - 주문 정보를 서버에 전송하는 함수.  - 주문자의 정보를 전송하면 서버에서 보내주는 주문 번호를 받아 화면에 출력한다.  - 장바구니에 담긴 케이크 인덱스 번호를 전송하고 함수를 종료한다.  //메뉴 정보 받아와야 함  for (int i = 0;i < MENU\_NUM;i++)  {  recv(hSocket, buf, BUF\_SIZE, 0); //메뉴 이름 길이 정보 전달 받음  send(hSocket, "fin", 4, 0); //수신 확인 메시지  strLen = atoi(buf); recLen = 0;  while (strLen != recLen)  {  recv(hSocket, &menu[i].menu[recLen++], 1, 0); //한글자씩 입력받음  }  send(hSocket, "fin", 4, 0); //수신 확인 메시지  recv(hSocket, buf, BUF\_SIZE, 0);//메뉴 가격 받아오기  menu[i].price = atoi(buf);  send(hSocket, "fin", 4, 0); //수신 확인 메시지  memset(buf, 0, BUF\_SIZE); //버퍼 초기화  }  - 메뉴들의 정보를 서버에서 받아와 저장한다.  if (input[1] == '\n' && input[0] >= '1' && input[0] < '1' + MENU\_NUM) //메뉴 선택 했을 시  {  if (shoplist.count == 9)  {  puts("장바구니가 꽉찼습니다."); continue;  }  AddMenu(menu, &shoplist, atoi(input) - 1);  printf("%s가 장바구니에 담겼습니다. \n", menu[atoi(input) - 1].menu);  Sleep(500);  }  - 메뉴 번호에 해당하는 번호가 입력되면 장바구니에 메뉴를 추가한다.  else if (input[1] == '\n' && input[0] == '0') //결제 선택했을 시  {  if (shoplist.count == 0)  {  puts("선택한 메뉴가 없습니다.");Sleep(500);  continue;  }  quit = Pay(&shoplist,menu); //결제 진행  if (quit == 0) continue;  SndList(&hSocket, shoplist);  Sleep(500);  }  - 결제를 진행하는 번호를 선택하면 결제를 진행한다.  - 선택된 메뉴가 없을 시 결제는 진행되지 않는다.  - 결제가 무사히 진행되면 서버에 주문 정보를 전송한다.  else if (input[2] == '\n' && input[0] == '-' && input[1] == '1') //장바구니 수정 선택했을 시  {  system("cls");  PrintList(shoplist,menu);  puts("===========================================");  printf("삭제하고 싶은 메뉴번호를 선택하세요(취소는 0)>");  memset(input, 0, sizeof(input));  fgets(input, sizeof(input), stdin);  if (input[1] == '\n' && input[0] >= '1' && input[0] <= '9')  Modifylist(&shoplist, atoi(input));  else if (input[1] == '\n' && input[0] == '0') continue;  else  {  puts("잘못된 입력입니다.");  }  Sleep(500);  }  - 메뉴 선택 취소를 선택했을 때 실행되는 코드이다.  - 장바구니 목록을 출력하고 삭제할 케이크의 목록 번호를 입력받는다. | | |
| **﻿프로젝트**  **소감** | 짧은 시간 안에 구현해 내기 위해 최대한 자세히 순서도를 그려내었고 그것을 바탕으로 빠르게 기틀을 잡아 원하는 최소한의 기능을 구현하기 위한 코드를 만들어낼 수 있었다. 실제로 일을 도와주는 곳의 메뉴얼을 바탕으로 구상하고 하나씩 구현해낼수록 재미도 느꼈고 눈으로 보이는 결과에 힘을 입어 더 많은 기능을 구현해보고자 할 수 있었다.  또한 2주간 리눅스 기반과 더불어 학습하였던 윈속 기반 프로그래밍에 익숙해질 수 있었던 좋은 기회였다.  앞으로의 프로젝트에서도 오늘의 경험을 기억해 일정을 체계적으로 관리하고 순서도도 최대한 자세히 그려내어 코드를 구현함에 있어서 막힘없이 진행할 수 있도록 해야겠다 | | | | |