

**과 목 : 프로그래밍 기초**

**과 제 명 : 개인 프로젝트**

**담 당 교 수 : 황 지 영 교수님**

**학 과 : 컴퓨터공학**

**학 년 : 3학년**

**학 번 : 2014315327**

**이 름 : 권현준**

**제 출 일 : 2016-06-07**

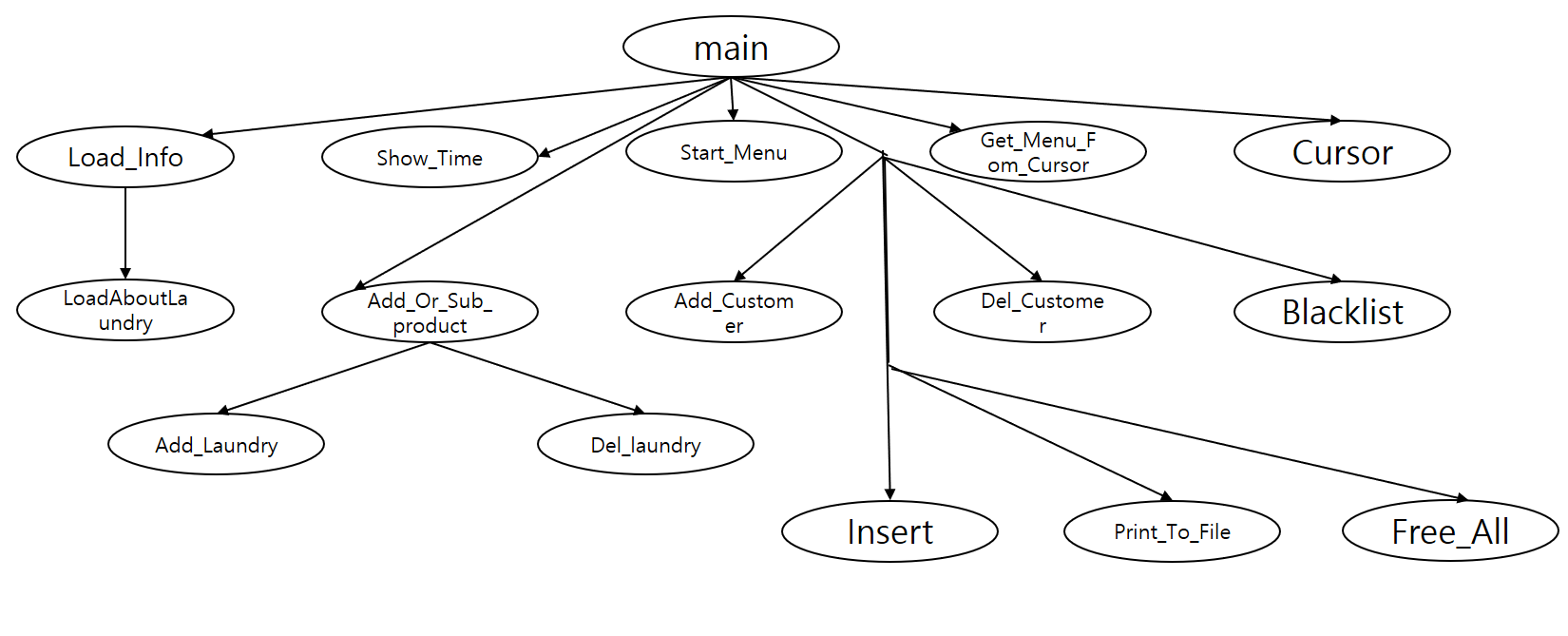
**1. 프로그램 설명 및 함수 각 프로토타입**

이번 프로젝트 과제로 구현한 프로그램은 세탁소에서 손님을 관리하는 프로그램입니다. 사용자 (세탁소 운영자)는 손님의 정보(이름, 휴대전화 번호, 주소, 메일 주소, 생년월일)를 받아 관리하며 손님을 추가 또는 삭제 할 수 있습니다. 또한 손님 개개인의 세탁물을 추가하거나 삭제하며 관리를 할 수 있으며 세탁물을 맡길 때 마다 3000원이 누적되고, 요금 미납자를 내림차순으로 출력하여 쉽게 볼 수 있게 하였습니다. 또한 요금을 결제하면 3000원이 차감되도록 하였습니다.

각 함수의 프로토타입은 다음과 같습니다.

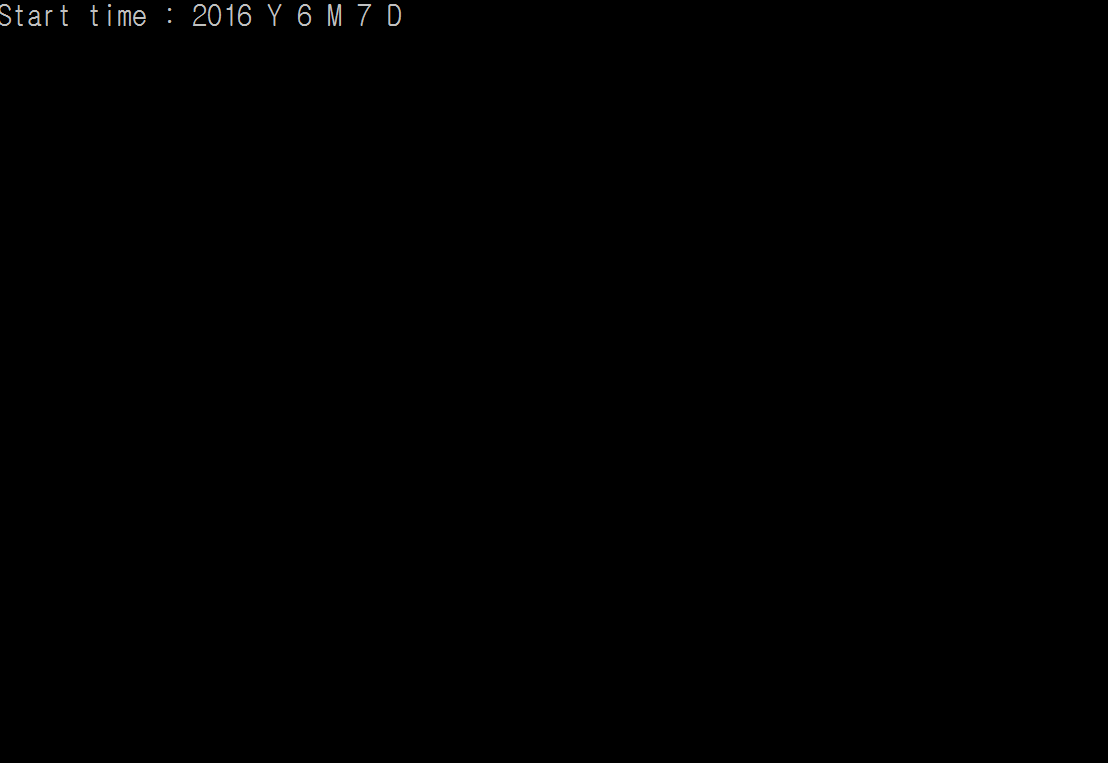
|  |
| --- |
| int Start\_Menu(void); // 메뉴를 결정하기 위한 함수  CUSTOMER \* Load\_Info(CUSTOMER \* head); // 손님의 정보를 읽어 오는 함수  void Insert(CUSTOMER \* x, CUSTOMER \* y); // 구조체 y를 x에 복사하는 함수  int Get\_Menu\_From\_Cursor(void); // 커서를 통해 메뉴를 선택하는 함수  void Add\_Customer(CUSTOMER \* head); // 손님을 추가하는 함수  void Del\_Customer(CUSTOMER\* head); // 손님을 삭제하는 함수  void LoadAboutLaundry(CUSTOMER \* temp); // 세탁물에 대한 정보를 읽어 오는 함수  void Add\_Or\_Sub\_Product(CUSTOMER \* temp, int day, int year, int mon); // 세탁물을 추가할지  또는 삭제할지 결정하는 함수  void Blacklist(CUSTOMER \* head); // 요금 미납자를 내림차순으로 최대 다섯명 까지 나타내는 함수  void Print\_To\_File(CUSTOMER \* temp); // 구조체 리스트의 내용을 파일에 출력하는 함수  void Free\_All(CUSTOMER \* temp); // 세탁물 정보 노드를 삭제하는 함수  void Add\_Laundry(CUSTOMER \* temp, int day, int year, int mon); // 세탁물을 추가하는 함수  void Del\_Laundry(CUSTOMER \* temp); // 세탁물을 삭제하는 함수  void Cursor(int x, int y); // 커서 함수  int Show\_Time(int \* day, int \* year, int \* mon); // 현재 시간을 보여주는 함수 |

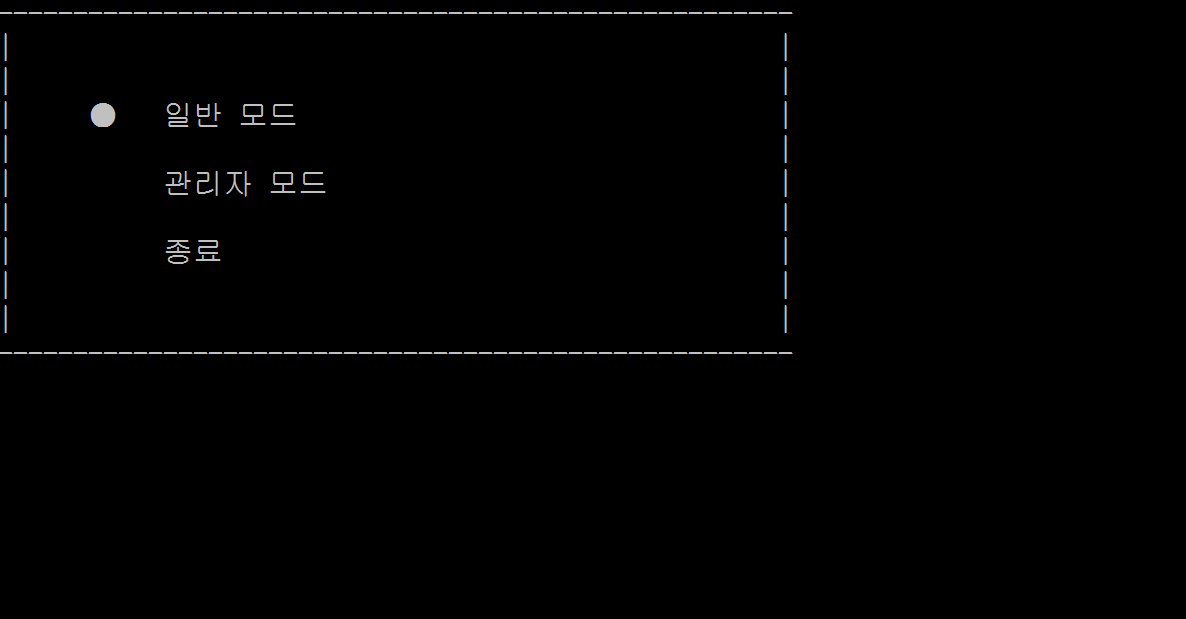
**2. 함수 간 호출 관계 도식화**



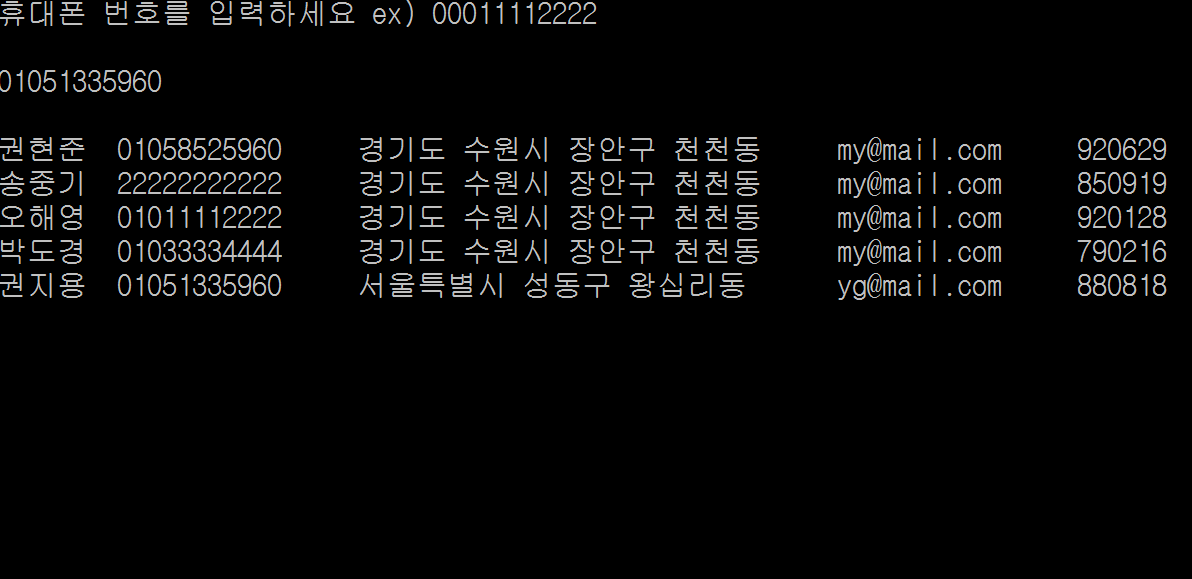
**3. 실행 화면**

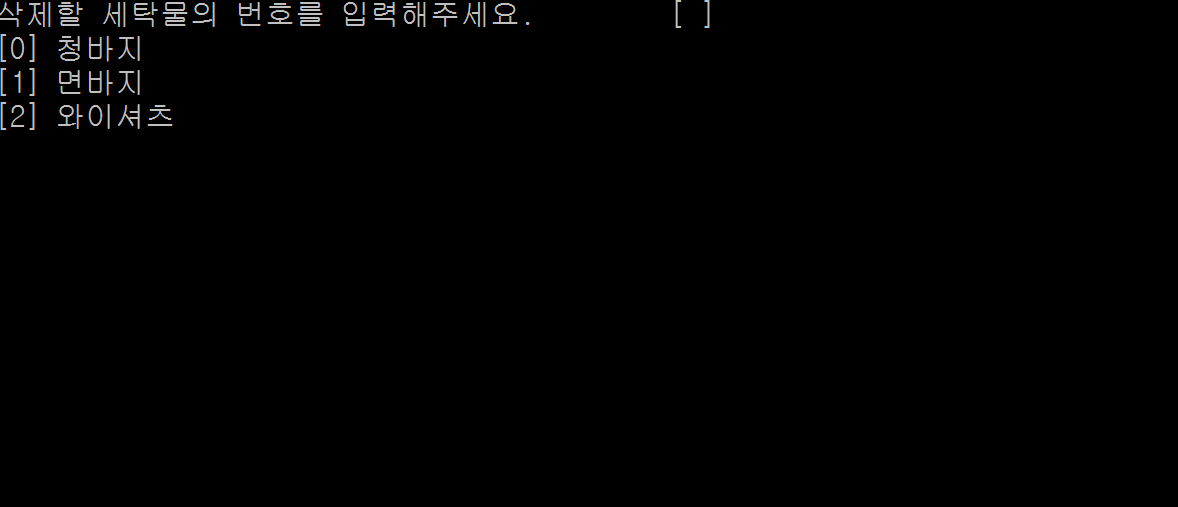
처음 프로그램을 실행하면 Show\_Time 함수를 통해 현재 날짜가 표시가 되고 그 다음 초기 메뉴 화면이 나오게 됩니다.



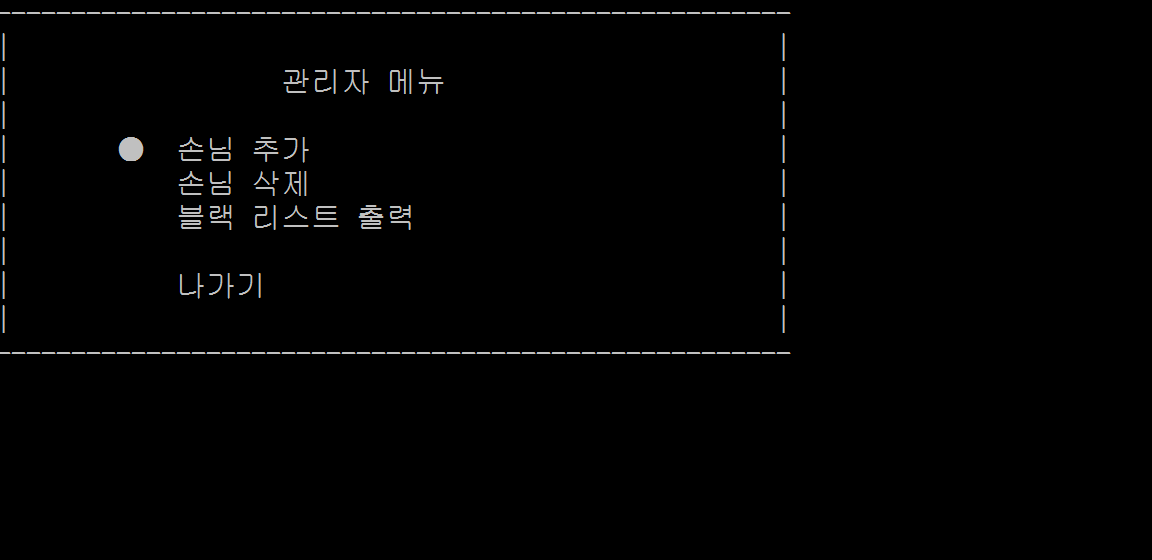


여기서 일반 모드를 선택하면 고객들의 정보가 출력되고 해당하는 고객의 휴대전화 번호를 입력하면, 해당 고객의 세탁물 정보를 볼 수 있습니다. 또한 세탁물의 추가와 삭제를 할 수 있습니다.





앞서 초기 메뉴화면에서 관리자 메뉴를 선택 시, 고객의 추가 및 삭제와 미납금에 따른 블랙리스트를 확인 할 수 있습니다.



**부록**

**Source code.**

|  |
| --- |
| **KWON.h**  /\*-------------------------------------------------------------------------------------------------------  gleentopia  Copyright (c) SungKyunKwan University  Author : Kwon Hyun Jun  Initial Creation : Wed, MAY, 18, 2016  Description : 세탁소에서 손님을 관리하는 프로그램입니다. 사용자는 손님의 정보 (이름, 휴대전화  번호, 주소, 메일, 생년월일)를 받아 관리하며 손님을 삭제 또는 추가 할 수 있습니다. 또한 손님 개개인의  세탁물 관리를 할 수 있으며 그에 따른 청구비용또한 관리할 수 있습니다.  --------------------------------------------------------------------------------------------------------\*/  // Header files inclusion  #include <stdio.h>  #include <windows.h>  #include <string.h>  #include <conio.h>  #include <time.h>  // Define constants  #define UP 72  #define DOWN 80  #define LEFT 75  #define RIGHT 77  #define ENTER 13  #define ESC 27  // Define structure  typedef struct \_LAUNDRY  {  char type[100];  int start\_day;  int end\_day;  struct \_CUSTOMER \* name;  }LAUNDRY;  typedef struct \_CUSTOMER  {  char name[100];  char phone[100];  char address[100];  char mail[100];  int unpaidbalance; // 미납금  int gender;  char idnumber[100];  int laundrynumber;  struct \_CUSTOMER \* prev;  struct \_CUSTOMER \* next;  LAUNDRY \*\* mylaundry; // Array that store LUNDRY address value  }CUSTOMER;  // Functions declare  int Start\_Menu(void); // Deciding menu  CUSTOMER \* Load\_Info(CUSTOMER \* head); // Read customer information  void Insert(CUSTOMER \* x, CUSTOMER \* y); // Copy structures  int Get\_Menu\_From\_Cursor(void); // Select menu from cursor  void Add\_Customer(CUSTOMER \* head);  void Del\_Customer(CUSTOMER\* head);  void LoadAboutLaundry(CUSTOMER \* temp);  void Add\_Or\_Sub\_Product(CUSTOMER \* temp, int day, int year, int mon);  void Blacklist(CUSTOMER \* head); // Search top 5 about unpaidbalance  void Print\_To\_File(CUSTOMER \* temp);  void Free\_All(CUSTOMER \* temp);  void Add\_Laundry(CUSTOMER \* temp, int day, int year, int mon);  void Del\_Laundry(CUSTOMER \* temp);  void Cursor(int x, int y); // Cursor  int Show\_Time(int \* day, int \* year, int \* mon); // Show present time |
| **main.c**  #include "KWON.h"  int main(void)  {  // Local variables declare  int selectmenu;  int getmenu;  int clock;  char c;  int day = 0;  int year = 0;  int mon = 0;  CUSTOMER \* head = NULL;  // Statements  head = Load\_Info(head); // Store customer's infomation in head  clock = Show\_Time(&day, &year, &mon); // Show present time  while (1)  {  fflush(stdin);  scanf("%c", &c);  if (c == '\n')  break;  } // If press enter, go to next statement  while (1)  {  selectmenu = Start\_Menu();  if (selectmenu == 1) // Normal mode  {  Add\_Or\_Sub\_Product(head, day, year, mon);  }  else if (selectmenu == 2) // Supervisor mode  {  while (1)  {  getmenu = Get\_Menu\_From\_Cursor();  if (getmenu == 1)  {  Add\_Customer(head);  break;  }  else if (getmenu == 2)  {  Del\_Customer(head);  }  else if (getmenu == 3)  {  Blacklist(head);  while (1)  {  fflush(stdin);  scanf("%c", &c);  if (c == '\n')  break;  }  break;  }  else  {  break;  }  }  }  else  {  Print\_To\_File(head);  Free\_All(head);  return 0;  }  }  } |
| **menu.c**  #include "KWON.h"  void Cursor(int x, int y)  {  COORD pos = { x, y };  SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), pos);  }  int Get\_Menu\_From\_Cursor(void)  {  // Local variables declare  char c;  char choice;  int menu = 1;  int x = 8, y = 4;  FILE \* customerfomat;  // Statements  system("cls");  if (!(customerfomat = fopen("customerfomat.txt", "r")))  {  printf("Error");  exit(101);  }  while (1) // Select menu  {  while ((c = fgetc(customerfomat)) != EOF)  printf("%c", c);  fseek(customerfomat, 0L, SEEK\_SET);  Cursor(x, y);  printf("●");  if (choice = getch())  {  switch (choice)  {  case UP:  y = y - 1;  menu = menu - 1;  break;  case DOWN:  y = y + 1;  menu = menu + 1;  break;  case ENTER:  return menu;  }  system("cls");  }  }  fclose(customerfomat);  return menu;  }  void Print\_To\_File(CUSTOMER \* temp)  {  // Local variables declare  char phonenum[100 + 4] = { 0, };  char text[] = ".txt";  FILE \* customerfile;  FILE \* loundryfile;  int i;  // Statements  customerfile = fopen("Customer.txt", "w");  if (customerfile == NULL)  {  printf("\"Customer.txt\"파일 오픈 에러!\n");  return;  }  while (temp != NULL)  {  fprintf(customerfile, "%s %s %s, %s %d %d %s\n", temp->name, temp->phone, temp->address, temp->mail, temp->unpaidbalance, temp->gender, temp->idnumber);  for (i = 0; i < (100 + 4); i++) // Initialize phonenum array  phonenum[i] = 0;  strcpy(phonenum, temp->phone);  strcat(phonenum, text);  // Using phonenum array, create  loundryfile = fopen(phonenum, "w");  if (loundryfile == NULL)  {  printf("\"%s\"파일 오픈 에러\n", phonenum);  return;  }  for (i = 0; i < temp->laundrynumber; i++)  {  fprintf(loundryfile, "%s %d %d", temp->mylaundry[i]->type, temp->mylaundry[i]->start\_day, temp->mylaundry[i]->end\_day);  if (i == ((temp->laundrynumber) - 1))  {  i++;  break;  }  fprintf(loundryfile, "\n");  }  fclose(loundryfile);  temp = temp->next;  }  fclose(customerfile);  return;  }  void Free\_All(CUSTOMER \* temp)  {  // Local variables declare  int i;  CUSTOMER\* temp2 = temp;  // Statements  while (temp2 != NULL)  {  temp2 = temp->next;  for (i = 0; i < temp->laundrynumber; i++)  free(temp->mylaundry[i]);  free(temp->mylaundry);  free(temp);  temp = temp2;  }  }  void LoadAboutLaundry(CUSTOMER \* temp)  {  // Local variables declare  char phone[16] = { 0 };  char arr[100];  char comp;  int line = 0;  int i;  FILE \* load;  // Statements  strcpy(phone, temp->phone);  strcat(phone, ".txt"); // Using phonenumber and strcat function, create ".txt" file  load = fopen(phone, "r");  if (load == NULL)  {  return;  }  while ((comp = fgetc(load)) != EOF)  {  if (comp == '\n')  {  line++;  }  }  line++;  temp->laundrynumber = line;  temp->mylaundry = (LAUNDRY\*\*)malloc(sizeof(LAUNDRY)\*line);  fseek(load, 0, SEEK\_SET);  for (i = 0; i < line; i++)  {  temp->mylaundry[i] = (LAUNDRY\*)malloc(sizeof(LAUNDRY));  temp->mylaundry[i]->name = temp;  fscanf(load, "%s", temp->mylaundry[i]->type);  fscanf(load, "%d", &temp->mylaundry[i]->start\_day);  fscanf(load, "%d", &temp->mylaundry[i]->end\_day);  }  // Read line number, and assign array that its size is line number  // and initialize variables in the array  fclose(load);  }  int Start\_Menu(void)  {  // Local variables declare  FILE \* file;  char c;  char choice;  int x = 6, y = 3;  int menu = 1;  // Statements  if (!(file = fopen("startfomat.txt", "r"))) // Starting fomat form  {  printf("Error");  exit(101);  }  while (1) // Deciding menu  {  system("cls");  while ((c = fgetc(file)) != EOF) // Getting starting fomat  printf("%c", c);  fseek(file, 0L, SEEK\_SET);  Cursor(x, y);  printf("●");  if (choice = getch())  {  switch (choice)  {  case UP:  y = y - 2;  menu = menu - 1;  break;  case DOWN:  y = y + 2;  menu = menu + 1;  break;  case ENTER:  return menu;  }  }  }  fclose(file);  }  CUSTOMER \* Load\_Info(CUSTOMER \* head)  {  // Local variables declare  FILE \* cus;  CUSTOMER \* temp;  CUSTOMER \* tail = 0;  CUSTOMER copy;  // Statements  if (!(cus = fopen("Customer.txt", "r")))  {  printf("Error");  return -1;  }  while (fscanf(cus, "%s %s %[^,]%\*c%s %d %d %s", copy.name, copy.phone, copy.address, copy.mail, &copy.unpaidbalance, &copy.gender, copy.idnumber) != EOF)  {// Using assignment suppression flag, %[^ ,] to read address until meet ','  if ((temp = (CUSTOMER\*)malloc(sizeof(CUSTOMER))) == NULL)  {  printf("Memory allocation error.");  return -1;  }  Insert(temp, &copy); // Insert copy data into temp  temp->laundrynumber = 0;  LoadAboutLaundry(temp);  temp->next = NULL; // Using linked list  if (head == NULL)  {  head = temp;  head->prev = NULL;  }  else  {  temp->prev = tail;  tail->next = temp;  }  tail = temp;  }  fclose(cus);  return head;  }  void Insert(CUSTOMER \* x, CUSTOMER \* y)  {  // Statements  strcpy(x->name, y->name);  strcpy(x->phone, y->phone);  strcpy(x->address, y->address);  strcpy(x->mail, y->mail);  x->unpaidbalance = y->unpaidbalance;  x->gender = y->gender;  strcpy(x->idnumber, y->idnumber);  }  int Show\_Time(int \* day, int \* year, int \* mon)  {  // Local variables declare  int day\_gap;  int starttime;  int finishtime;  FILE \* file;  time\_t now = 0;  struct tm \* t = 0; // A function that brings present time in sec unit  // Statements  time(&now); // A time function brings time imformation to time data type variable 'now'  t = localtime(&now); // Store time information using localtime function in variable 't'  \*year = t->tm\_year + 1900; // and calculate it year, month, day unit  \*mon = t->tm\_mon + 1;  \*day = t->tm\_mday;  printf("Start time : %d Y %d M %d D", \*year, \*mon, \*day);  starttime = ((\*year) \* 10000) + ((\*mon) \* 100) + (\*day); // Store time information ex)20160519  file = fopen("time.txt", "a+");  fscanf(file, "%d", &finishtime);  day\_gap = starttime - finishtime; // Subract last quit time of program and present program stating time  return day\_gap;  } |
| **customer.c**  #include "KWON.h"  void Add\_Customer(CUSTOMER \* head)  {  // Local variables declare  CUSTOMER \* temp;  CUSTOMER \* newcustomer;  int cmp;  if ((newcustomer = (CUSTOMER\*)malloc(sizeof(CUSTOMER))) == NULL)  {  printf("Memory assignment error!");  exit(101);  }  system("cls");  // Store nwe customer's information temporarily in newcustomer  printf("이름을 입력하세요");  scanf("%s", newcustomer->name);  printf("핸드폰 번호를 입력하세요");  scanf("%s", newcustomer->phone);  printf("주소를 입력하세요");  scanf("%s", newcustomer->address);  printf("이메일 주소를 입력하세요");  scanf("%s", newcustomer->mail);  printf("성별 을 입력하세요 (남자 1, 여자 2)");  scanf("%d", &newcustomer->gender);  printf("주민번호를 입력하세요");  scanf("%s", newcustomer->idnumber);  temp = head; // As Temp = head, start from head  while (1)  {  cmp = strcmp(temp->name, newcustomer->name);  if (cmp >= 0) // If temp' name is longer than new's name  {  if (temp->prev == NULL) // When temp goes to last order, newcustomer's information is stored  {  newcustomer->prev = NULL;  newcustomer->next = temp;  head = newcustomer;  return;  }  newcustomer->prev = temp->prev;  temp->prev->next = newcustomer;  temp->prev = newcustomer;  newcustomer->next = temp;  return;  }  else {  if (temp->next == NULL) // If there are no data to compare  {  temp->next = newcustomer;  newcustomer->prev = temp;  newcustomer->next = NULL;  return;  }  temp = temp->next; // Using temp->next pointer, keep moving and searching  }  }  }  void Del\_Customer(CUSTOMER \* head)  {  // Local variables declare  CUSTOMER \* temp;  char tempphone[12] = { 0 };  // Statements  temp = head;  system("cls");  printf("정보를 삭제할 회원의 핸드폰번호를 알려주세요\n"); // Because people's phone number is harder to be same than people's name  scanf("%s", tempphone);  while (1)  {  if ((strcmp(temp->phone, tempphone)) == 0)  {  printf("%s 회원님의 정보를 삭제합니다.\n", temp->name);  if (temp != head)  { // Using linked list principle, swap present value of temp->next and temp->prev->next  temp->prev->next = temp->next;  temp->next->prev = temp->prev;  free(temp); // Delete temp node  temp = NULL;  printf("회원 정보가 정상적으로 삭제되었습니다.\n");  return;  }  else  {  head = temp->next;  head->prev = NULL;  free(temp);  return;  }  }  else  {  if (temp->next == NULL)  {  printf("원하시는 회원의 정보가 존재하지 않습니다.");  return;  }  temp = temp->next; // Keep linked list going temp->next  }  }  }  void Blacklist(CUSTOMER \* head)  {  // Local variables declare  int count = 0;  int tempmoney[5] = { 0, };  char tempname[5][40] = { 0, }; // To store name temporarily  CUSTOMER \* temp;  // Statements  temp = head;  system("cls");  while (temp->next != NULL)  {  // Search highest unpaidbalance and name who has it storing tempmoney[0], tempname[0]  if (temp->unpaidbalance >= tempmoney[count])  {  strcpy(tempname[count], temp->name);  tempmoney[count] = temp->unpaidbalance;  }  temp = temp->next;  }  // And next, store decending order  for (count = 1; count < 5; count++)  {  temp = head;  while (temp != NULL)  {  if (temp->unpaidbalance >= tempmoney[count])  {  if (strcmp(tempname[0], temp->name) != 0 && strcmp(tempname[1], temp->name) != 0  && strcmp(tempname[2], temp->name) != 0 && strcmp(tempname[3], temp->name) != 0  && strcmp(tempname[4], temp->name) != 0)  { // Except same name, and decending order and if the unpaidbalance is same amount,  // store it alphabetic and length order  strcpy(tempname[count], temp->name);  tempmoney[count] = temp->unpaidbalance;  }  }  temp = temp->next;  }  }  printf("\n 미납금 TOP 5\n");  for (count = 0; count < 5; count++)  {  printf("[%d] %s님 : %d 원\n", count + 1, tempname[count], tempmoney[count]);  }  return;  } |
| **laundry.c**  #include "KWON.h"  void Add\_Or\_Sub\_Product(CUSTOMER \* head, int day, int year, int mon)  {  // Local variables declare  char phonenumber[100 + 4] = { 0 };  char txt[] = ".txt";  int find = 0;  CUSTOMER \* temp = head;  char c = "";  int i = 0;  char laundryaddordel = "";  char k = "";  // Statements  system("cls");  printf("휴대폰 번호를 입력하세요 ex) 00011112222\n");  Cursor(0, 4);  while (temp != NULL)  {  printf("%s\t", temp->name);  printf("%s\t", temp->phone);  printf("%s\t", temp->address);  printf("%s\t", temp->mail);  printf("%s\n", temp->idnumber);  temp = temp->next;  }  Cursor(0, 2);  scanf("%s", phonenumber);  temp = head;  while (temp != NULL) // Searching customer  {  if (strcmp(phonenumber, temp->phone) == 0)  {  find = 1;  break;  }  temp = temp->next;  }  if (find == 0)  {  system("cls");  printf("손님정보를 찾을수 없습니다 \n");  while (1)  {  fflush(stdin);  scanf("%c", &c);  if (c == '\n')  break;  }  return;  }  else  {  system("cls");  printf("-------------맡긴 세탁물-------------\n\n");  printf("종류\t맡긴 날짜\t찾아갈 날짜 \n\n");  for (i = 0; i < temp->laundrynumber; i++)  {  printf("%s\t%d\t%d", temp->mylaundry[i]->type, temp->mylaundry[i]->start\_day, temp->mylaundry[i]->end\_day);  if (i == (temp->laundrynumber) - 1)  {  i++;  break;  }  printf("\n");  }  printf("\n\n--------------------------------------\n");  printf("추가하시려면 1을 삭제하시려면 2를 눌러주세요.");  laundryaddordel = getch();  if (laundryaddordel == '1')  {  free(temp->mylaundry);  Add\_Laundry(temp, day, year, mon);  }  else  Del\_Laundry(head);  system("cls");  getchar();  printf("금액을 지불하셨습니까? Y/N ");  scanf\_s("%c", &k);  if (k == 'Y')  temp->unpaidbalance = 0;  }  return;  }  void Add\_Laundry(CUSTOMER \* temp, int day, int year, int mon)  {  // Local variables declare  char phone[16] = { 0 };  char arr[100];  char li;  FILE \* load;  int line = 0;  int i;  char type[100];  // Statements  strcpy(phone, temp->phone);  strcat(phone, ".txt");  load = fopen(phone, "a");  if (load == NULL)  {  return;  }  system("cls");  printf("추가할 옷의 종류를 입력해주세요\n\n");  printf("와이셔츠, 러닝셔츠, 청바지, 면바지, 패딩, 코트\n\n [ ]\b\b\b\b\b\b\b\b\b\b\b");  scanf("%s", type);  fprintf(load, "\n%s\t%d%0.2d%0.2d\t%d%0.2d%0.2d", type, year, mon, day, year, mon, day + 3);  temp->unpaidbalance = temp->unpaidbalance + 3000; // Recalulate unpaidbalance  fclose(load);  load = fopen(phone, "r");  while ((li = fgetc(load)) != EOF)  {  if (li == '\n')  {  line++;  }  }  line = line + 2;  temp->laundrynumber = line - 1;  temp->mylaundry = (LAUNDRY\*\*)malloc(sizeof(LAUNDRY)\*(line));  fseek(load, 0, SEEK\_SET);  for (i = 0; i<line; i++) // Renew information  {  temp->mylaundry[i] = (LAUNDRY\*)malloc(sizeof(LAUNDRY));  temp->mylaundry[i]->name = temp;  fscanf(load, "%s", temp->mylaundry[i]->type);  fscanf(load, "%d", &temp->mylaundry[i]->start\_day);  fscanf(load, "%d", &temp->mylaundry[i]->end\_day);  }  fclose(load);  }  void Del\_Laundry(CUSTOMER \* temp)  {  // Local variables declare  int i;  int c;  int a = 0;  LAUNDRY \*\* laundry;  int line = 0;  // Statements  system("cls");  printf("삭제할 세탁물의 번호를 입력해주세요.\n");  for (i = 0; i < temp->laundrynumber; i++)  {  printf("[%d] ", i);  printf("%s\t\n", temp->mylaundry[i]->type);  } // Print all laundry  Cursor(45, 0);  printf("[ ]\b\b");  scanf("%d", &c);  temp->mylaundry[i] = NULL;  temp->laundrynumber = temp->laundrynumber - 1;  laundry = (LAUNDRY\*\*)malloc(sizeof(LAUNDRY)\*(temp->laundrynumber)); // Allocate laundry  for (i = 0; i< temp->laundrynumber; i++)  {  if (temp->mylaundry[i] != NULL)  {  laundry[i] = (LAUNDRY\*)malloc(sizeof(LAUNDRY));  }  }  for (i = 0; i< temp->laundrynumber + 1; i++)  {  if (i != c)  {  strcpy(laundry[a]->type, temp->mylaundry[i]->type);  laundry[a]->start\_day = temp->mylaundry[i]->start\_day;  laundry[a]->end\_day = temp->mylaundry[i]->end\_day;  a++;  }  }  free(temp->mylaundry);  temp->mylaundry = laundry;  return;  } |