C 프로그래밍Ⅱ

기말 프로젝트

컴퓨터공학과

20145150

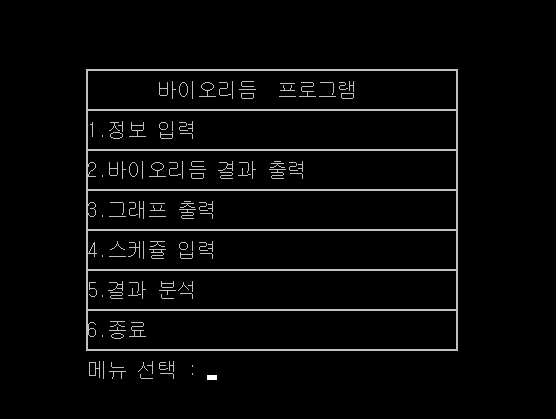
이상욱

만들게 된 계기

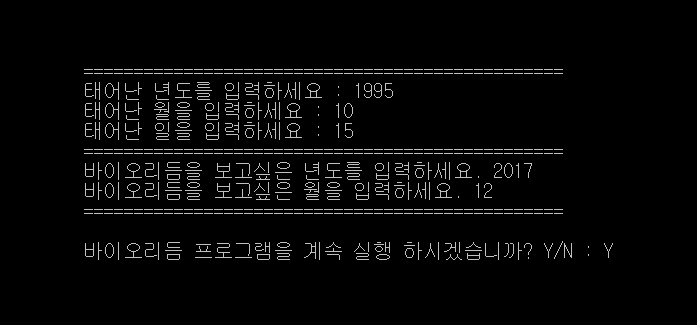
아침에 일어나면 심심풀이로 운세를 보고 그날 하루가 어떨지 예상하는 것은 자주 있는 일이다. 그런데 이번에 바이오리듬 코딩을 수업시간에 배우면서 스케줄을 입력을 하면 스케줄을 바이오 리듬과 비교하여 이 스케줄을 실행하는데 알맞은 리듬을 가지고 있는지 판단해서 알려주고 만약 그 스케줄이 리듬과 맞지 않다면 다른 스케줄을 추천하는 형식의 프로그램이 있으면 그 날 그 것을 인지하고 스케줄을 이행하는데 도움이 될 수 있겠다고 생각했다. 이것을 활용 하여 아침에 일과를 시작하기 전에 알려주면 좋을 것 같다는 생각으로 프로젝트를 만들어 보았다.

용도

바이오리듬을 활용 하여 그날의 바이오리듬을 확인하고 자신의 스케줄과 그에 필요한 리듬을 입력하여 그 날에 그 스케줄을 실행 했을 경우 어떠한지 알려주는 프로그램이다.

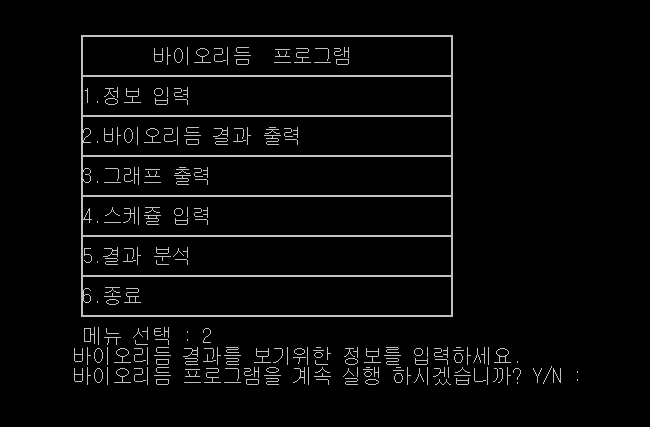


프로그램을 실행하면 처음으로 나오는 화면이다 메뉴를 통해 원하는 정보를 입력하고 출력 받게 하였다. 메뉴 기반 프로그램으로 사용자가 원할 때까지 프로그램을 반복해서 실행한다.

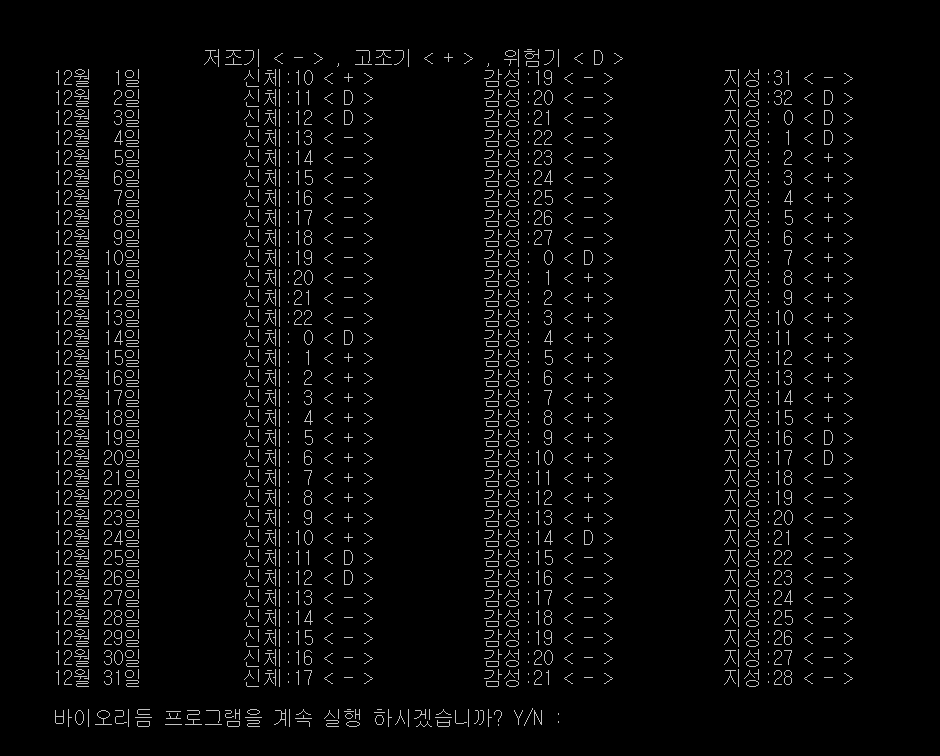


1번을 입력하면 생년월일과 바이오리듬을 보고 싶은 날짜를 입력 받는다.

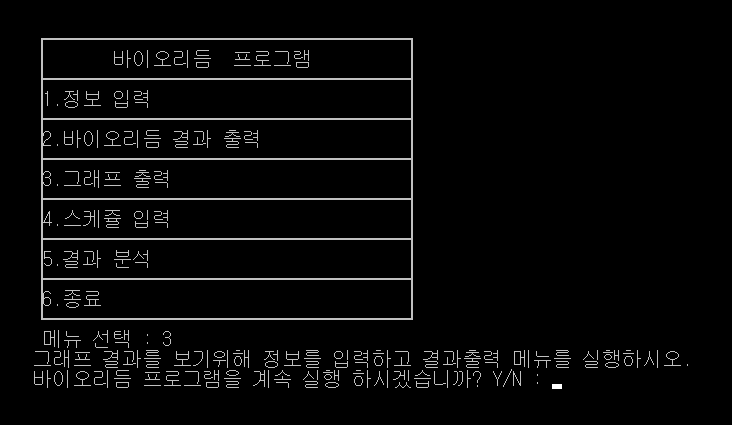
바이오리듬 결과를 보기 위해 개인의 생년월일을 입력하고 자신이 보고 싶은 바이오리듬의 년도와 월을 입력 받는다.



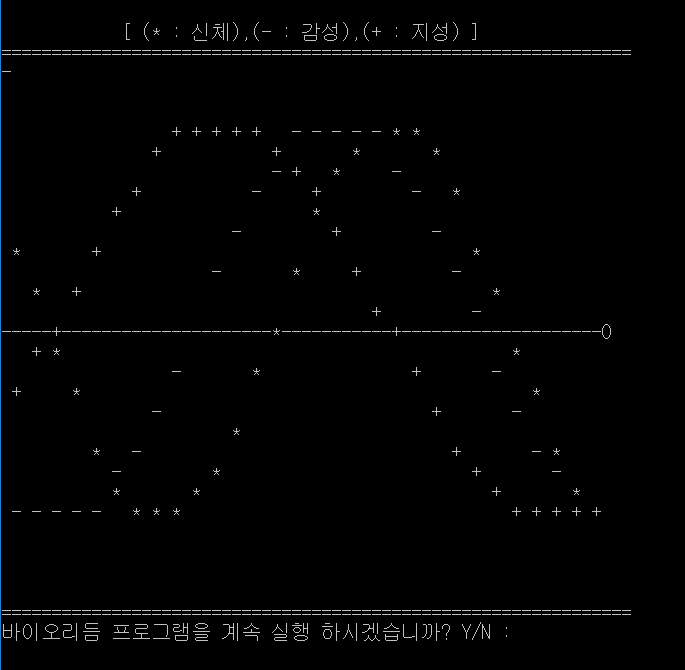
1번을 실행 했을 경우 2번을 실행 할 수 있다. 1번을 실행하지 않았을 경우 다음과 같은 결과가 출력 된다.



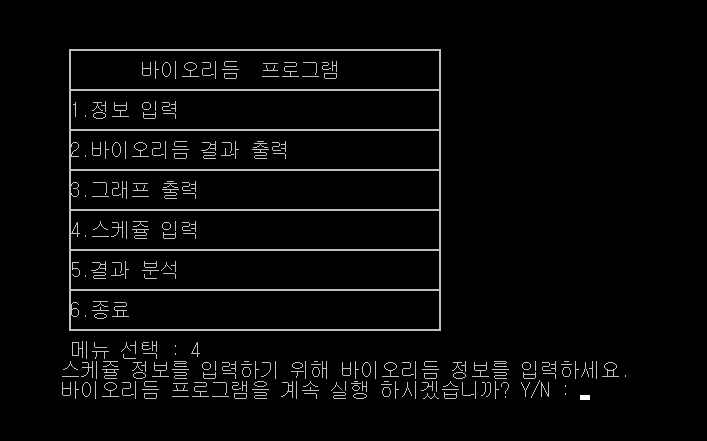
1번을 실행해 정보를 입력하고 2번을 실행했을 경우 입력한 달에 일마다의 신체 감성 지성 리듬을 출력해 바이오리듬 결과를 볼 수 있다.



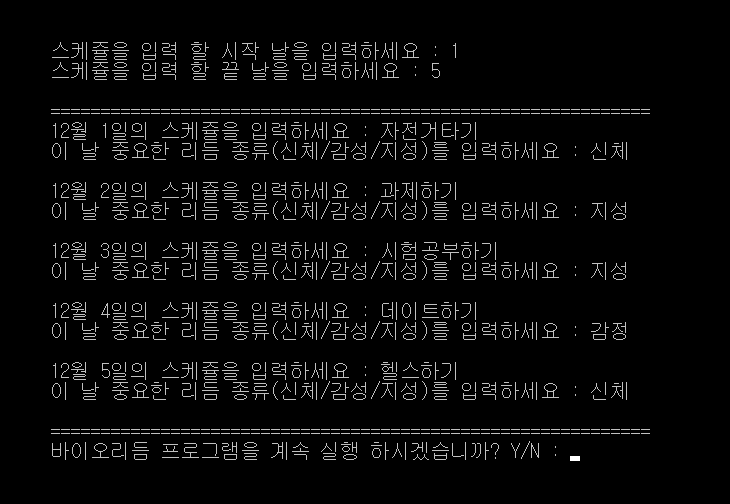
1번 2번을 실행했을 경우 3번을 실행 할 수 있다. 1번 2번을 실행하지 않고 3번을 실행했을 경우 실행화면 이다.



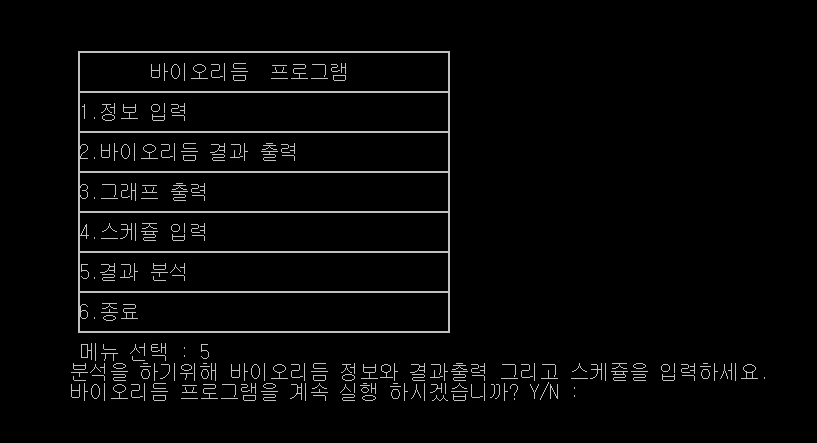
1번을 이용해 정보를 입력 받고 2번을 실행해 결과가 나왔을 경우 그 결과로 그래프로 출력해 바이오리듬 결과를 한눈에 그래프로 볼 수 있다.



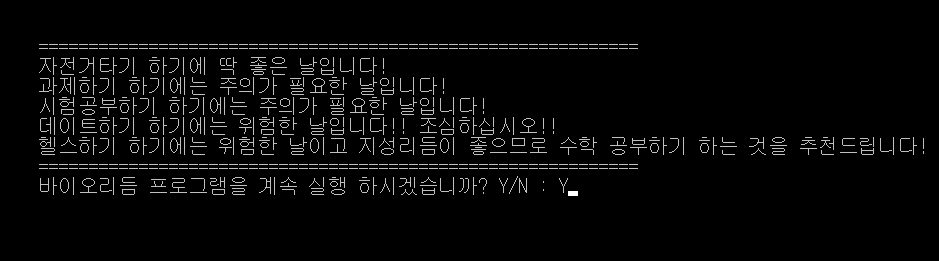
1번을 실행 했을 경우 4번을 실행 할 수 있다. 1번을 실행 하지 않았을 경우 출력 화면이다.



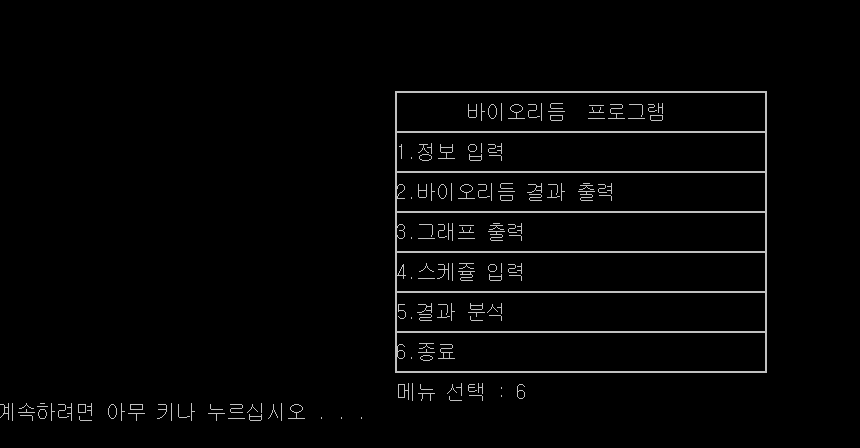
한달 모두를 입력 받으면 입력 받는 내용이 너무 많으므로 보고 싶은 날의 시작 날과 끝 날을 입력 받게 만들었다. 그리고 스케줄과 그 스케줄에 필요한 리듬을 입력 받는다.



1번 2번 4번을 실행했을 경우 5번을 실행 할 수 있다. 1번 2번 4번을 실행 하지 않았을 경우 실행화면 이다.



스케줄에 필요한 리듬과 바이오리듬 결과를 분석하여 그날 그 스케줄을 이행하는데 어떠할지를 출력 받는다. 고조기일 경우는 실행하기 좋은 날이라고 알려주고 위험기일 경우에는 주의가 필요하다고 알려준다. 저조기일 경우는 그 스케줄을 이행하면 손해가 있을 수 있으므로 다른 리듬 두개 중 고조기인 리듬을 필요로 하는 활동을 추천하고 다른 두 리듬도 고조기가 없을 경우 위험하다고 경고를 한다.



6번은 프로그램을 종료한다.

기타

모든 출력화면은 헤더파일 windos.h과 gotoXY함수를 사용해 명령프롬포트의 좌표를 파악해 원하는 위치의 좌표대로 이동하여 출력 하였다.

프로젝트를 마치고

스케줄을 입력했을 경우 프로그램에서 알아서 그 스케줄에 필요한 리듬을 파악해서 바이오리듬 결과와 비교해서 결과 분석을 하게 하고 싶었지만 그렇게 하려면 입력 받을 수 있는 스케줄을 모두 배열에 입력하고 그에 따른 리듬도 다 같은 index의 다른 배열에 입력을 해야 되기에 어려움이 있었다. 그래서 리듬도 따로 입력을 받았는데 만약 현존하는 운세 보는 프로그램처럼 바이오리듬 프로그램을 만들려면 스케줄을 비교하려면 스케줄마다 리듬을 다 입력해줘야 할 것 같다. 결과를 파일로 저장하고 그 결과를 원할 때 마다 볼 수 있게 하고 싶었지만 프로그램 용도상 파일로 저장해서 볼 정도로 사람들이 자주 사용하는 것은 아니고 날마다 바뀌기 때문에 제외했다. 그리고 CMD로 디자인을 할 수 있는 것이 출력화면의 위치 이동 정도 밖에 할 수 없다는 것이 아쉬웠다.

/\* 프로젝트 이름 : Project1.c

작성자 : 이상욱

학번 : 20145150

학과 : 컴퓨터 공학과

작성날짜 : 2017.12.01 ~ 04

프로젝트 목적 : 바이오리듬을 사용해서 자신이 원하는 요일에 스케쥴을 실행할때 그 스케쥴에 필요한 리듬이 그 날에 어떠한지 분석하여 알려주는 프로그램

\*/

#define \_CRT\_SECURE\_NO\_WARNINGS //scanf오류로 사용 하였습니다.

#include <stdio.h>

#include <windows.h> //gotoxy사용

#include<math.h> //sin함수 사용

#include<time.h> //랜덤값을 받기 위해 사용

#include<stdlib.h> //랜덤함수 사용

#include<string.h> //strcmp사용

void gotoxy(int x, int y); //출력위치 변환 함수

void Biorhythm\_Information(int \*born\_year,int \*born\_month,int \*born\_day,int\* year, int\*month); //바이오리듬계산을 위한 정보입력 함수

int Biorhythm\_Result(char arr[][3], int \*born\_year, int \*born\_month, int \*born\_day, int\* year, int\*month, int month\_size); //바이오리듬 결과 출력 밑 그래프 출력을 위한 총 일 수 반환 함수

int total\_days(int year, int month, int day); //이 날까지의 일수 계산 함수

char bio\_status(int v, int mod); //고조기 저조기 위험기를 계산하기 위한 함수

void Biorhythm\_Graph(int total\_day); //그래프 출력 함수

void Schedule\_rhythm(char arr\_rythm[][20], char arr\_scehdule[][20],int month\_size , int month, int start , int end); //스케쥴을 입력받기 위한 함수

void Biorhythm\_Analyses(char arr\_rythm[][20], char arr\_scehdule[][20], char arr[][3] ,int start , int end); //스케쥴과 바이오리듬을 분석하여 결과를 출력하는 함수

int main(int argc, char argv[]) {

char arr[31][3] = { 0 }; //바이오리듬 배열

char arr\_rythm[31][20] = { 0 }; //스케쥴에 필요한 리듬배열

char arr\_scehdule[31][20] = { 0 }; //스케쥴 배열

char \*Menu[] = { "1.정보 입력","2.바이오리듬 결과 출력 ","3.그래프 출력","4.스케쥴 입력 ","5.결과 분석","6.종료" }; //menu 포인터 배열

char ch = 'Y'; //프로그램 계속 해서 실행할 건지 여부

int menu; //메뉴 선택

int born\_year, born\_month, born\_day = 0; //생년월일

int year, month, day = 0; //바이오리듬 보고싶은 날짜

int total\_day; //총일수

int months\_size[] = { 31,28,31,30,31,30,31,31,30,31,30,31 }; //매달 일수가 다르므로 배열로 생성해서 처리

int runCount1, runCount2 , runCount3; //실행한 메뉴를 확인하기 위한 변수

int start, end; //탐색을 원하는 시작 날과 끝 날

while (ch == 'Y' || ch =='y')

{

system("cls"); //cmd 초기화

gotoxy(40, 10); //출력 위치 변경 후 메뉴 출력

printf("┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┓\n");

gotoxy(40, 11);

printf("┃ 바이오리듬 프로그램 ┃\n");

gotoxy(40, 12);

printf("┣━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┫\n");

gotoxy(40, 13);

printf("┃%-36s┃\n", Menu[0]);

gotoxy(40, 14);

printf("┣━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┫\n");

gotoxy(40, 15);

printf("┃%-36s┃\n", Menu[1]);

gotoxy(40, 16);

printf("┣━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┫\n");

gotoxy(40, 17);

printf("┃%-36s┃\n", Menu[2]);

gotoxy(40, 18);

printf("┣━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┫\n");

gotoxy(40, 19);

printf("┃%-36s┃\n", Menu[3]);

gotoxy(40, 20);

printf("┣━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┫\n");

gotoxy(40, 21);

printf("┃%-36s┃\n", Menu[4]);

gotoxy(40, 22);

printf("┣━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┫\n");

gotoxy(40, 23);

printf("┃%-36s┃\n", Menu[5]);

gotoxy(40, 24);

printf("┗━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┛\n");

gotoxy(40, 25);

printf(" 메뉴 선택 : ");

scanf("%d", &menu);

if (menu == 1)

{

system("cls");

Biorhythm\_Information(&born\_year,&born\_month,&born\_day,&year,&month); //생년월일 및 보고싶은 년월 입력

runCount1 = 0; //실행했는지 확인

runCount1++;

}

else if (menu == 2) //1번을 실행했을 경우에만 실행가능

{

if (runCount1 == 1)

{

system("cls");

total\_day = Biorhythm\_Result(arr, &born\_year, &born\_month, &born\_day, &year, &month, months\_size[month - 1]); //바이오리듬 결과보고 그래프 출력을 위한 총 일수값 반환 받기

runCount2 = 0; //실행했는지 확인

runCount2++;

}

else

{

gotoxy(40, 26);

printf("바이오리듬 결과를 보기위한 정보를 입력하세요. \n");

gotoxy(40, 27);

}

}

else if (menu == 3)

{

if (runCount2 == 1 && runCount1 == 1) //1번 2번을 실행했을 경우에만 실행가능

{

system("cls");

Biorhythm\_Graph(total\_day); //바이오리듬 그래프보기

}

else

{

gotoxy(40, 26);

printf("그래프 결과를 보기위해 정보를 입력하고 결과출력 메뉴를 실행하시오. \n");

gotoxy(40, 27);

}

}

else if (menu == 4)

{

if (runCount1 == 1) //1번이 실행했을 경우에만 실행가능

{

system("cls");

gotoxy(35, 12);

printf("스케쥴을 입력 할 시작 날을 입력하세요 : ");

scanf("%d", &start);

gotoxy(35, 13);

printf("스케쥴을 입력 할 끝 날을 입력하세요 : ");

scanf("%d", &end);

Schedule\_rhythm(arr\_rythm, arr\_scehdule, months\_size[month - 1], month , start , end); //스케쥴 입력

runCount3 = 0;

runCount3++;

}

else

{

gotoxy(40, 26);

printf("스케쥴 정보를 입력하기 위해 바이오리듬 정보를 입력하세요. \n");

gotoxy(40, 27);

}

}

else if (menu == 5)

{

if (runCount2 == 1 && runCount1 == 1 && runCount3 == 1) //메뉴 1번 2번 4번 실행 했을 경우에만 실행가능

{

system("cls");

Biorhythm\_Analyses(arr\_rythm, arr\_scehdule, arr, start , end);

}

else

{

gotoxy(40, 26);

printf("분석을 하기위해 바이오리듬 정보와 결과출력 그리고 스케쥴을 입력하세요. \n");

gotoxy(40, 27);

}

}

else if (menu == 6) //종료

{

break;

}

else //이외의 것을 입력하였을 경우

{

gotoxy(40, 26);

printf("잘 못 입력하셨습니다. \n");

gotoxy(40, 27);

}

printf("바이오리듬 프로그램을 계속 실행 하시겠습니까? Y/N : "); //2번 실행하고 N눌러서 종료시 오류

scanf(" %c", &ch);

}

return 0;

}

void gotoxy(int x, int y) //출력위치 변환 함수

{

COORD Pos = { x - 1 , y - 1 };

SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE),Pos);

}

void Biorhythm\_Information(int \*born\_year, int \*born\_month, int \*born\_day, int\* year, int\*month) //생년월일 및 바이오리듬 보고싶은 년월 입력

{

gotoxy(35, 12);

printf("================================================\n");

gotoxy(35, 13);

printf("태어난 년도를 입력하세요 : ");

scanf("%d", born\_year);

gotoxy(35, 14);

printf("태어난 월을 입력하세요 : ");

scanf("%d", born\_month);

gotoxy(35, 15);

printf("태어난 일을 입력하세요 : ");

scanf("%d", born\_day);

gotoxy(35, 16);

printf("================================================\n");

gotoxy(35, 17);

printf("바이오리듬을 보고싶은 년도를 입력하세요. ");

scanf("%d", year);

gotoxy(35, 18);

printf("바이오리듬을 보고싶은 월을 입력하세요. ");

scanf("%d", month);

gotoxy(35, 19);

printf("================================================\n");

printf("\n");

gotoxy(35, 21);

}

int Biorhythm\_Result(char arr[][3], int \*born\_year, int \*born\_month, int \*born\_day, int \*year, int \*month, int month\_size) //바이오 리듬 결과보기

{

int i;

int k = 9;

int Sum\_Bornday, Sum\_Nowday; //1년1월1일 부터 태어난날 바이오리듬 보고싶은 날 까지의 일수를 저장 할 변수

int total; //총일수 변수

int physical, emotion, intellect; //신체,감성,지성리듬 변수

int temp; //총일수 값이 변화 하므로 반환에 쓰일 총일수 임시저장 변수

Sum\_Bornday = total\_days(\*born\_year, \*born\_month, \*born\_day); //1년1월1일부터 생년월일까지의 일수

Sum\_Nowday = total\_days(\*year, \*month, 1); //1년1월1일부터 바이오리듬 보고싶은 날 까지의 일수

total = Sum\_Nowday - Sum\_Bornday; //둘을 빼면 태어난 날부터 바이오리듬 보고싶은 날 까지의 일수가 나옴

temp = total; //변수 저장

gotoxy(30, 8);

printf("저조기 < - > , 고조기 < + > , 위험기 < D > \n");

printf("\n");

for (i = 1 ; i <= month\_size; i++)

{

physical = total % 23; //신체리듬의 주기는 23일

emotion = total % 28; //감성리듬의 주기는 28일

intellect = total % 33; //지성리듬의 주기는 33일

gotoxy(15, k++);

printf("%d월 %2d일", \*month, i); //월일 출력

arr[i][0] = bio\_status(physical, 23); //고조기 저조기 위험기 판단 후에 기호 입력

arr[i][1] = bio\_status(emotion, 28); //고조기 저조기 위험기 판단 후에 기호 입력

arr[i][2] = bio\_status(intellect, 33); //고조기 저조기 위험기 판단 후에 기호 입력

printf(" \t 신체:%2d < %c > ", physical, arr[i][0]); //결과출력

printf(" \t 감성:%2d < %c > ", emotion, arr[i][1]);

printf(" \t 지성:%2d < %c > \n", intellect, arr[i][2]);

total++; //다음 일로 증가

}

printf("\n");

gotoxy(15, ++k);

return temp; //총 일 수 반환

}

int total\_days(int year, int month, int day) //총일 수 계산 함수

{

int months[] = {31,28,31,30,31,30,31,31,30,31,30,31}; //각 월별로 가지는 일수 배열

int i;

int total= 0;

total = (year-1)\*365 + (year-1)/4 - (year-1)/100 + (year-1)/400; //년마다의 일수와 겹치는 윤년 빼기

if(!(year%4) && (year%100) || !(year%400)) //겹치는 윤년 빼고 진짜 윤년 일 경우에 일수 하나 증가

{

months[1]++;

}

for(i=0;i<month-1;i++) //월별로 일수 증가

{

total += months[i];

}

total += day; //일수 만큼 일수 증가

return total;

}

char bio\_status ( int v , int mod )

{

char result;

double r;

r = sin(v\*3.141592 / (mod / 2.0)); //sin((바이오리듬 값 \* 원주율) / (주기/2))

if (r < -0.2) //-0.2보다 낮을 경우 저조기

{

result = '-';

}

else if ((-0.2 <= r) && (r <= 0.2)) //-0.2 와 0.2 사이 일 경우 위험기

{

result = 'D';

}

else

{

result = '+'; //0.2이상일 경우 고조기

}

return result; //결과 기호 반환

}

void Biorhythm\_Graph(int total\_day) //그래프 출력 함수

{

double r[32], r1[32], r2[32]; //

int physical, emotion, intellect;

int i, y;

for (i = 1; i <= 30 ; i++)

{

physical = total\_day % 23;

emotion = total\_day % 28;

intellect = total\_day % 33;

r[i] = sin(physical\*3.141592 / (23 / 2.0));

r1[i] = sin(emotion\*3.141592 / (28 / 2.0));

r2[i] = sin(intellect\*3.141592 / (33 / 2.0));

total\_day++;

}

gotoxy(1, 5);

printf(" [ (\* : 신체),(- : 감성),(+ : 지성) ] \n");

gotoxy(1, 6);

printf("===============================================================\n");

for (i = 0 ; i <= 60 ; i++) //중앙지점 이 선보다 아래일 경우 그날 리듬이 나쁨 위 일경우 그날 리듬이 좋음

{

gotoxy(i, 20);

printf("-");

}

printf("0");

for (i = 1 ; i <= 30 ; i++)

{

y = 20 - 10 \* r[i];

gotoxy(i \* 2, y); //신체 리듬 그래프

printf("\*");

y = 20 - 10 \* r1[i];

gotoxy(i \* 2, y); //감성 리듬 그래프

printf("-");

y = 20 - 10 \* r2[i];

gotoxy(i \* 2, y); //지성 리듬 그래프

printf("+");

}

gotoxy(1, 34);

printf("===============================================================\n");

gotoxy(1, 35);

}

void Schedule\_rhythm(char arr\_rythm[][20] , char arr\_scehdule[][20], int month\_size , int month ,int start , int end) //스케쥴입력 받는 함수

{

int i;

int k = 16;

gotoxy(35, 14);

if (end > month\_size) { //보고싶은 끝날이 이번 달 일수 보다 많으면 값이 이상하게 나오므로 이번달 일수로 맞춤

printf("끝 날짜가 이번 달 일수 보다 많으므로 끝 날짜를 이번 달 일수로 교체하겠습니다. \n");

end = month\_size;

}

gotoxy(35, 15);

printf("============================================================\n");

for (i = start; i <= end; i++) //스케쥴을 입력받고 그에 필요한 리듬 입력받음

{

gotoxy(35,k++);

printf("%d월 %d일의 스케쥴을 입력하세요 : ", month , i);

scanf("%s", arr\_scehdule[i]);

gotoxy(35, k++);

printf("이 날 중요한 리듬 종류(신체/감성/지성)를 입력하세요 : ");

scanf("%s", arr\_rythm[i]);

gotoxy(35, k++);

printf("\n");

}

gotoxy(35, k++);

printf("============================================================\n");

gotoxy(35, k++);

}

void Biorhythm\_Analyses(char arr\_rythm[][20], char arr\_scehdule[][20], char arr[][3], int start , int end) //결과 분석함수

{

int i;

int k = 13;

int num;

int runCount1 = 0, runCount2 = 0; //실행했는지 확인 하는 변수

char \*physical[] = { "자전거 타기" , "축구 하기" , "달리기" , //감성,지성리듬이 좋고 신체리듬이 안 좋을 경우 랜덤으로 값을 정해 이 배열의 그 랜덤값 위치에 있는 행동을 추천함

"등산하기" , "농구 하기" , "야구 하기" ,

"헬스 하기" , "줄넘기 하기" , "족구 하기" };

char \*emotion[] = { "음악 듣기" , "영화 보기", "카페 가기" , //신체.지성리듬이 좋고 감성리듬이 안 좋을 경우 랜덤으로 값을 정해 이 배열의 그 랜덤값 위치에 있는 행동을 추천함

"산책 하기" , "데이트 하기","술 한잔 하기",

"여행 가기", "편지 쓰기", "친구들 만나기" };

char \*intellect[] = { "영어 공부하기", "수학 공부하기", "국어 공부하기", //신체.감성리듬이 좋고 지성리듬이 안 좋을 경우 랜덤으로 값을 정해 이 배열의 그 랜덤값 위치에 있는 행동을 추천함

"C언어 공부하기", "자바 공부하기", "알고리즘 공부하기",

"과제 하기" , "자격증 공부하기", "과학서적 읽기" };

srand((int)time(NULL)); //시드 값을 계속 변화시켜줌

gotoxy(30, 12);

printf("============================================================\n");

for (i = start; i <= end; i++) //보고싶은 날별로 필요한 리듬과 그 날 실제 리듬을 비교 후 분석결과 출력

{

if (strcmp(arr\_rythm[i],"신체") == 0) //신체 리듬이 필요한 날일 경우

{

if (arr[i][0] == '-') //신체리듬이 필요하지만 신체리듬이 필요한 일을 하기에 저조기 일경우

{

if (arr[i][1] == '+') //신체리듬이 안 좋지만 감성리듬이 좋을 경우

{

num = rand() % 9; //랜덤 값 받기

gotoxy(30, k++);

printf("%s 하기에는 위험한 날이고 감성리듬이 좋으므로 %s 하는 것을 추천드립니다! \n", arr\_scehdule[i], emotion[num]); //위의 emotion배열의 랜덤 값 index의 있는 행동 추천

runCount1++; //실행 확인

}

if (arr[i][2] == '+') //신체리듬이 안 좋지만 지성리듬이 좋을 경우

{

num = rand() % 9; //랜덤 값 받기

gotoxy(30, k++);

printf("%s 하기에는 위험한 날이고 지성리듬이 좋으므로 %s 하는 것을 추천드립니다! \n", arr\_scehdule[i], intellect[num]); //위의 intellect배열의 랜덤 값 index의 있는 행동 추천

runCount2++;

}

if(runCount1 == 0 && runCount2 == 0) //신체리듬도 저조기이고 다른 리듬도 고조기가 없을 경우

{

gotoxy(30, k++);

printf("%s 하기에는 위험한 날입니다!! 조심하십시오!! \n",arr\_scehdule[i]);

}

runCount1 = 0; //실행 확인 햇으니 초기화시켜줌

runCount2 = 0; //실행 확인 햇으니 초기화시켜줌

}

else if (arr[i][0] == 'D') //신체리듬이 필요하지만 신체리듬이 필요한 일을 하기에 위험기 일경우

{

gotoxy(30, k++);

printf("%s 하기에는 주의가 필요한 날입니다! \n", arr\_scehdule[i]);

}

else //신체리듬이 필요하고 고조기 일경우

{

gotoxy(30, k++);

printf("%s 하기에 딱 좋은 날입니다! \n", arr\_scehdule[i]);

}

}

else if(strcmp(arr\_rythm[i], "감성") == 0) //감성 리듬이 필요한 날일 경우

{

if (arr[i][1] == '-') //감성리듬이 필요하지만 감성리듬이 필요한 일을 하기에 저조기 일경우

{

if (arr[i][0] == '+') //신체리듬은 좋을경우 신체리듬을 필요로 하는 활동을 추천

{

num = rand() % 9; //랜덤 값 받기

gotoxy(30, k++);

printf("%s 하기에는 위험한 날이고 신체리듬이 좋으므로 %s 하는 것을 추천드립니다! \n", arr\_scehdule[i], physical[num]); //위의 physical배열의 랜덤 값 index의 있는 행동 추천

runCount1++; //실행 확인

}

if (arr[i][2] == '+') //지성리듬은 좋을경우 지성리듬을 필요로 하는 활동을 추천

{

num = rand() % 9; //랜덤 값 받기

gotoxy(30, k++);

printf("%s 하기에는 위험한 날이고 지성리듬이 좋으므로 %s 하는 것을 추천드립니다! \n", arr\_scehdule[i], intellect[num]); //위의 intellect배열의 랜덤 값 index의 있는 행동 추천

runCount2++; //실행 확인

}

if (runCount1 == 0 && runCount2 == 0) //감성리듬도 저조기이고 다른 리듬도 고조기가 없을 경우

{

gotoxy(30, k++);

printf("%s 하기에는 위험한 날입니다!! 조심하십시오!! \n", arr\_scehdule[i]);

}

runCount1 = 0;

runCount2 = 0;

}

else if (arr[i][1] == 'D') //감성리듬이 필요하지만 신체리듬이 필요한 일을 하기에 위험기 일경우

{

gotoxy(30, k++);

printf("%s 하기에는 주의가 필요한 날입니다! \n", arr\_scehdule[i]);

}

else //감성리듬이 필요하고 고조기 일경우

{

gotoxy(30, k++);

printf("%s 하기에 딱 좋은 날입니다! \n", arr\_scehdule[i]);

}

}

else //지성 리듬이 필요한 날일 경우

{

if (arr[i][2] == '-') //지성리듬이 필요하지만 지성리듬이 필요한 일을 하기에 저조기 일경우

{

if (arr[i][0] == '+') //신체리듬은 좋을경우 신체리듬을 필요로 하는 활동을 추천

{

num = rand() % 9; //랜덤 값 받기

gotoxy(30, k++);

printf("%s 하기에는 위험한 날이고 신체리듬이 좋으므로 %s 하는 것을 추천드립니다! \n", arr\_scehdule[i], physical[num]); //위의 physical배열의 랜덤 값 index의 있는 행동 추천

runCount1++; //실행 확인

}

if (arr[i][1] == '+') //감성리듬은 좋을경우 감성리듬을 필요로 하는 활동을 추천

{

num = rand() % 9; //랜덤 값 받기

gotoxy(30, k++);

printf("%s 하기에는 위험한 날이고 감성리듬이 좋으므로 %s 하는 것을 추천드립니다! \n", arr\_scehdule[i], emotion[num]); //위의 emotion배열의 랜덤 값 index의 있는 행동 추천

runCount2++; //실행 확인

}

if (runCount1 == 0 && runCount2 == 0) //지성리듬도 저조기이고 다른 리듬도 고조기가 없을 경우

{

gotoxy(30, k++);

printf("%s 하기에는 위험한 날입니다!! 조심하십시오!! \n", arr\_scehdule[i]);

}

runCount1 = 0;

runCount2 = 0;

}

else if (arr[i][2] == 'D') //지성리듬이 필요하지만 신체리듬이 필요한 일을 하기에 위험기 일경우

{

gotoxy(30, k++);

printf("%s 하기에는 주의가 필요한 날입니다! \n", arr\_scehdule[i]);

}

else //감성리듬이 필요하고 고조기 일경우

{

gotoxy(30, k++);

printf("%s 하기에 딱 좋은 날입니다! \n", arr\_scehdule[i]);

}

}

}

gotoxy(30, k++);

printf("============================================================\n");

gotoxy(30, k++);

}