

# Introduction to Computer and Lab

## Homework #6

**Due date: Mar 19, 2016**

**학번: 201404051**

이름: 정 용 석

## 1. 조합값 구하기

### 1.1 Solution

주어진 3 개의 함수, `get_interger()`, `combination()`, `factorial()` 함수를 이용하여 결과 값을 출력하는 문제이다. 일단 각 함수의 목적과 기능을 살펴보면, 일단 `get_interger()` 함수는 간단하게 사용자로부터 입력 받은 수를 반환하는 함수이다. 그래서 반환 형을 보면 `int` 로 말 그대로 사용자에게 입력 받은 `int` 형 자료를 반환 한다. `Factorial()` 함수 같은 경우는 일단 반환 형은 `long long` 이고 인자는 `int n` 으로서, 이를 그대로 `n!(factorial)`를 계산하여 반환해 주면 된다. `Factorial` 같은 경우 값이 엄청나게 커지기 때문에 `long long` 자료형을 이용하여 반환하게 되어있다. 마지막으로 `Comination()` 함수를 살펴보면, 마찬가지로 반환 형은 `long long` 그리고 인자는 `int n` 과 `r` 을 받는 함수로서, `n` 과 `r` 의 `factorial` 의 조합을 이용하여 계산된 값을 반환해주는 함수이다. 이 조합에 대한 공식은 수업시간에 나온 공식을 그대로 이용하였다. 이 3 개의 함수를 사용하여 나온 결과 값을 출력하는 함수로서 `computeCombination()` 함수가 쓰였다. 이는 간단하게 위의 `combination` 함수로 나온 값을 프로그램화 한 함수로서 정리해주는 함수로 보기 쉽다.

## 1.2. Source code

[illegible]

```

        for (i = 1; i <= n; i++)
            result *= i;
        return result;
}

//정수 n과 r의 조합값을 반환하는 함수
long long combination(int n, int r)
{
    ///////////////////////////////////
    return factorial(n) / (factorial(r)*factorial(n - r));
}

// 결과 값 출력
void computeCombination()
{
    int n = get_integer();
    int r = get_integer();

    printf("C(%d, %d) = %lld\n", n, r, combination(n, r));
}

```

### 1.3. Result (snapshot)

```

20
2
C(20, 2) = 190

20
10
C(20, 10) = 184756

3
2
C(3, 2) = 3

15
4
C(15, 4) = 1365

13
13
C(13, 13) = 1

30
2
C(30, 2) = -1

^CPress any key to continue . .

```

## 2 완전수 구하기

### 2.1. Solution

2 부터 10000 까지의 수 중 완전수를 구하는 문제이다. 간단하게, 숫자를 2 부터 10000 까지 늘리면서 자기 수를 제외한 약수를 모두 더하여 자기 자신이 되는 수를 찾으면 된다. 이 문제를 위해 2 개의 함수가 사용된다. 첫번째가 int n 이 완전수인지 아닌지를 판별하는 checkPerfect()와 이를 이용하여 2 부터 10000 까지의 모두 완전수를 구하는 findPerfectNumber()함수이다.

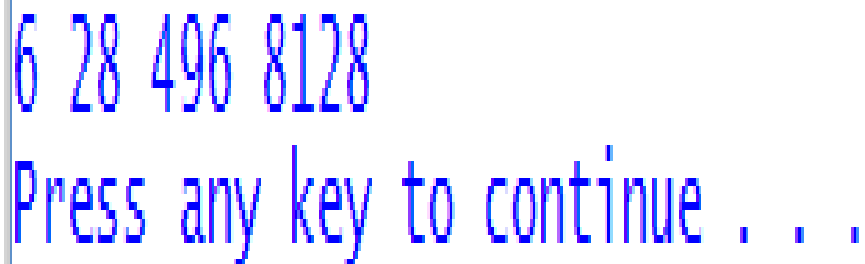
### 2.2. Source code

```
// 완전수이면 1, 아니면 0을 반환
int checkPerfect(int n)
{
    //////////////////////////////////////
    int sum = 0;
    int i;
    for (i = 1; i < n; i++)
        if (n % i == 0)
            sum += i;

    if (sum == n)
        return 1;
    else
        return 0;
}

// 2~10000 완전수를 출력
void findPerfectNumber()
{
    int i;
    for (i = 2; i <= 10000; i++)
        if (checkPerfect(i))
            printf("%d ", i);
    printf("\n");
}
```

### 2.3. Result (snapshot)



```
6 28 496 8128
Press any key to continue . . .
```

### 3 10 친화수 구하기

#### 3.1. Solution

2 와 1000 사이의 모든 친화수를 찾는 문제로서 2 번과 비슷하게 일단 친화수의 여부를 판단하는 `checkFrindNumber` 함수와 친화수를 찾는 `findFrindNumber` 함수로 나뉜다.

`checkFrindNumber` 함수는 일단 `int` 형 인자를 2 개 받는 함수인데, 말 그대로 2 개의 정수가 서로 친화수인지만 확인해주면 된다. 따라서 각 정수의 약수를 찾아서 각 약수의 총합이 서로의 값과 동일하면 친화수가 된다. 이를 이용하여 `findFrindNumber`에서는 2 부터 1000 까지의 수를 이중 반복 문을 통하여 가능한 경우의 수를 확인, 친화수를 찾게 된다.

#### 3.2. Source code

```
// 두 수 n1과 n2가 친화수이면 1, 아니면 0
int checkFrindNumber(int n1, int n2)
{
    //////////////////////////////////////
    int sum1 = 0; //n1의 약수의 총합
    int sum2 = 0; //n2의 약수의 총합
    int i;
    for (i = 1; i < n1; i++)
        if (n1 % i == 0)
            sum1 += i;
    for (i = 1; i < n2; i++)
        if (n2 % i == 0)
            sum2 += i;

    //n1과 n2의 약수의 총합이 서로의 값과 같으면 1
    if ((sum1 == n2) && (sum2 == n1))
        return 1;
    else
        return 0;
}

// 2~1000 친화수를 찾는다
void findFrindNumber()
{
    int i, j;
    for (i = 2; i <= 10000; i++)
    {
        for (j = i + 1; j <= 10000; j++)
            if (checkFrindNumber(i, j))
                printf("(%d, %d) ", i, j);
        }
    printf("\n");
}
```

### 3.3. Result (snapshot)

```
(220, 284) (1184, 1210) (2620, 2924) (5020, 5564) (6232, 6368)
Press any key to continue . . .
```

## 4. 숫자 야구 게임

### 4.1. Solution

일반적으로 알고 있는 야구 게임으로서 사용자와 컴퓨터가 서로 대결하는 구도이다. 일단 게임이 시작되면, 프로그램은 자동적으로 3 개의 난 수를 만든다. 이 3 개의 수는 중복되지 않게 구현되어야 하고, 사용자에게 입력에 따라서 스트라이크, 볼, 아웃 여부를 출력해준다. 따라서 스트라이크와 볼을 체크할 수 있는 변수를 만들어주고, 총 9 Round 로 진행되므로 이를 체크해 주는 변수 또한 만들어 주었다. 자리를 일치 여부를 판단하기 위해, 컴퓨터가 만든 난 수와 사용자가 입력한 3 개의 수 각각의 변수를 따로 선언해주었고, 이를 순차적으로 비교하여 스트라이크와 볼을 if 문을 통하여 각각 할당한다. 프로그램이 승패 여부는 9 라운드까지 가거나, 혹은 사용자가 3 스트라이크, 즉 모든 수와 위치를 맞추게 되면 끝나기 때문에 while 반복 문을 통하여 구현하였고, 반복 문의 끝에는 항상 변수를 초기화 해주어야 한다. 각각의 상황에 따른 출력 문 또한 다르기 때문에 if 문을 통하여 적절하게 편성해주어야 한다. 마지막으로 이 게임은 사용자가 n 을 누르기 전까지는 무한으로(사실 상 n 이 아닌 다른 것을 입력하여도 프로그램이 종료된다. 이는 사용자가 잘못된 입력 시에 따른 예외 상황을 따로 처리하지 않았다.) 프로그램이 실행이 되기 때문에, 이를 while 무한 반복 문을 통하여 처음에 구현하였는데, 문제가 사용자가 3 개의 수를 입력하고 마지막에 누르는 enter 키에 있었다. Std input 버퍼에 enter 가 남아있어서, 다음 게임을 위해 사용자가 y 나 n 을 누르기 전에 이미 scanf 를 통해 enter 키가 입력되는 것을 확인할 수 있었다. 따라서 Std input 버퍼에 남아있는 enter 키를 없애기 위해 게임이 종료되고 getchar 함수를 통하여 버퍼에 남아있는 입력을 모두 없애 주고 시작해야 했다.

### 4.2. Source code

```
// 숫자 야구게임
void DigitBaseballGame()
{
    //////////////////////////////////////
    char start;           //시작여부
    int count = 1;        //라운드 횟수
    int n1, n2, n3;       //컴퓨터 난수 3개
    int i1, i2, i3;       //사용자 입력 3개
    int strike = 0;       //스트라이크
    int ball = 0;         //볼

    while (1) {
```

```

printf("Play game ? <y/n> :");
scanf("%c", &start);
if (start != 'y') {
    printf("Program Closes\n");
    break;
}

srand((unsigned)time(NULL));
while (1) //난수 3개 생성, 중복되면 다시
{
    n1 = rand() % 10;
    n2 = rand() % 10;
    n3 = rand() % 10;
    if (n1 != n2 && n1 != n3 && n2 != n3)
        break;
}
//테스트용 출력
printf("%d %d %d\n", n1, n2, n3);

while (1)
{
    printf("**** BASEBALL GAME ****   ROUND: %d\n", count++);
    printf("USER: ");
    scanf("%d %d %d", &i1, &i2, &i3);

    //if문을 각각의 입력 수 차례로 비교해서 스트라이크와 볼을 구분해야한다.
    //if문 하나에 다 넣게 되면 모든 경우의 수를 뽑아낼 수 없다.
    if (n1 == i1)
        strike++;
    else if (n1 == i2 || n1 == i3)
        ball++;

    if (n2 == i2)
        strike++;
    else if (n2 == i1 || n2 == i3)
        ball++;

    if (n3 == i3)
        strike++;
    else if (n3 == i1 || n3 == i2)
        ball++;

    //스트라이크 3개면 사용자의 승리 및 게임 종료
    if (strike == 3) {
        printf("USER WINNER !!!\n");
        printf("The numbers are %d %d %d\n", n1, n2, n3);
        //다음 게임을 위한 변수 초기화
        strike = 0;
        ball = 0;
        count = 1;
        break;
    }
}

```

```

//스트라이크와 볼이 모두 있을 때,
if (strike != 0 && ball != 0)
    printf("COMPUTER: %d STRIKE, %d BALL !!!!\n", strike, ball);
//스트라이크만 있을 때
else if (strike != 0 && ball == 0)
    printf("COMPUTER: %d STRIKE !!!!\n", strike);
//볼만 있을 때
else if (ball != 0 && strike == 0)
    printf("COMPUTER: %d BALL !!!!\n", ball);
//하나도 못 맞추면 OUT
else
    printf("COMPUTER: O U T !!!!\n");

//라운드가 끝나면 컴퓨터 승리 및 정답 제공, 프로그램 종료
if (count == 10)
{
    printf("\nCOMPUTER WINNER !!!!\n");
    printf("The numbers are %d %d %d\n", n1, n2, n3);
    //변수 초기화
    strike = 0;
    ball = 0;
    count = 1;
    break;
}
printf("\n");
strike = 0;
ball = 0;
}
//버퍼에 있는 ENTER 키를 없애기 위한 함수
getchar();
}
}

```

### 4.3. Result (snapshot)

```

Play game ? <y/n> :y
7 4 5
**** BASEBALL GAME ****   ROUND: 1
USER: 4 5 7
COMPUTER: 3 BALL !!!!

**** BASEBALL GAME ****   ROUND: 2
USER: 7 5 4
COMPUTER: 1 STRIKE, 2 BALL !!!!

**** BASEBALL GAME ****   ROUND: 3
USER: 1 2 3
COMPUTER: O U T !!!!

**** BASEBALL GAME ****   ROUND: 4
USER: 4 1 2
COMPUTER: 1 BALL !!!!

**** BASEBALL GAME ****   ROUND: 5
USER: 9 8 5
COMPUTER: 1 STRIKE !!!!

**** BASEBALL GAME ****   ROUND: 6
USER: 3 5 7
COMPUTER: 2 BALL !!!!

**** BASEBALL GAME ****   ROUND: 7
USER: 2 3 4
COMPUTER: 1 BALL !!!!

**** BASEBALL GAME ****   ROUND: 8
USER: 4 5 6
COMPUTER: 2 BALL !!!!

**** BASEBALL GAME ****   ROUND: 9
USER: 2 8 9
COMPUTER: O U T !!!!

COMPUTER WINNER !!!!
The numbers are 7 4 5
Play game ? <y/n> :n
Program Closes
Press any key to continue . . .

```

## 5 바이오 리듬 구하기

### 5.1. Solution

사용자가 생년월일 및 바이오리듬의 년도와 월을 입력하였을 때, 바이오리듬의 결과를 출력하는 프로그램으로, 총 4 개의 함수를 구현하여 해결하여야 한다. 첫 번째가 윤년을 판단하는 `checkLeapYear()` 함수로 이는 전 과제에서 많이 구현한 코드를 그대로 사용하였다. 두 번째로 주어진 달이 며칠까지 있는지 판별하는 함수로 이 또한 전 과제 중 달력을 출력하는 함수에서 `switch` 문으로 구현한 코드를 있는 그대로 사용하였다. 생년월일과 입력한 년 월까지의 총 날짜 수를 구하는 `countDate` 함수 같은 경우도 전 과제의 달력 문제와 비슷하기에 구현도 편하다. 한가지 문제가 될 만한 것은 생년월일은 일수까지 입력되기 때문에 그 달의 일 수에서 생년월일의 일 수를 일단 따로 계산해야 된다는 점이다. 그리고 생년월일의 다음 월부터 12 월까지의 일 수를 구하고, 그 다음해부터 입력 받은 해까지의 총 일수를 전부 따로 구했다. 물론 다른 방법도 있지만 이렇게 해야 덜 헛갈린 것 같다.

의아했던 점이 `printBioRhythm` 과 `computeBioRhythm` 함수 였던 것이, 이 2 개의 함수의 목적을 바꿔야 하는 것이 아닌가 한다. 하지만 예제에 나와 있듯이 구성과 맞게 구현하기 위해 `printBioRhythm` 함수에서 각각의 바이오 리듬을 계산하고 출력까지 담당하게 하였고, `computeBioRhythm` 함수에서는 사용자에게 입력을 도맡게 하게 하였다.

### 5.2. Source code

```
// 주어진 연도가 윤년인지 아닌지 판별하는 함수
int checkLeapYear(int year)
{
    //////////////////////////////////////
    if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0))
        return 1;
    else
        return 0;
}

// 주어진 달이 며칠까지 있는지 판별하는 함수
int checkMonth(int month, int leap_year)
{
    switch (month) {
    case 4:
    case 6:
    case 9:
    case 11:
        return 30;
        break;
    case 2: //윤년이면 29일, 보통은 28일
        if (checkLeapYear(leap_year))
            return 29;
    }
```



```

        else
            return 28;
        break;
case 1:
case 3:
case 5:
case 7:
case 8:
case 10:
case 12:
    return 31;
    break;
}
}

```

// 생년월일과 년과 월을 입력하였을 때 총 날짜 수 계산

```

int countDate(int birth_year, int birth_month, int birth_day, int year, int month)
{
    //////////////////////////////////////
    int i;
    int totalDays = 0;
    //일단 그 달의 일수 더하기
    totalDays += checkMonth(birth_month, birth_year) - birth_day;
    //다음달 부터 12월까지 일수 더하기
    for (i = birth_month + 1; i <= 12; i++)
        totalDays += checkMonth(i, birth_year);
    //다음해부터 입력된 해 전까지의 일수 더하기
    for (i = birth_year + 1; i < year; i++)
    {
        if (checkLeapYear(i))
            totalDays += 366;
        else
            totalDays += 365;
    }
    return totalDays;
}

```

// 총 날짜 수와 연과 월 입력시 바이오리듬 출력

```

void printBiorhythm(int date, int year, int month)
{
    //////////////////////////////////////
    char phys, ment, intel;          //신체, 정신, 지능 고조기, 저조기, 위험기 표현 변수
    int physP, mentP, intelP;        //신체, 정신, 지능 주기
    int i;
    int days = checkMonth(month, year);

    printf("Biorhythm Result:\n");
    printf("Low(-), High(*), Danger(D)\n");
    for (i = 1; i <= days; i++)
    {
        //신체, 정신, 지능 주기 계산
        physP = (date+ i - 1) % 23;

```

```

    mentP = (date + i - 1) % 28;
    intelP = (date + i - 1) % 33;

    //각각의 저조기, 고조기, 위험기 나누기
    if (physP >= 1 && physP <= 10)
        phys = '+';
    else if (physP >= 13 && physP <= 21)
        phys = '-';
    else if (physP == 0 || physP == 11 || physP == 12 || physP == 22)
        phys = 'D';

    if (mentP >= 1 && mentP <= 12)
        ment = '+';
    else if (mentP >= 15 && mentP <= 26)
        ment = '-';
    else if (mentP == 0 || mentP == 13 || mentP == 14 || mentP == 27)
        ment = 'D';

    if (intelP >= 1 && intelP <= 15)
        intel = '+';
    else if (intelP >= 18 && intelP <= 31)
        intel = '-';
    else if (intelP == 0 || intelP == 16 || intelP == 17 || intelP == 32)
        intel = 'D';

    printf("%dM %2dD P: %3d %cWt", month, i, physP, phys);
    printf("M: %3d %cWt", mentP, ment);
    printf("I: %3d %cWn", intelP, intel);
}

// 바이올리듬 계산하기
void computeBiorhythm()
{
    int birth_year, birth_month, birth_day;
    int year, month, date;

    printf("Input a birthdate (ex: 1983 5 21)Wn");
    scanf("%d %d %d", &birth_year, &birth_month, &birth_day);

    printf("Insert wanted year and month. (ex: 2004 11)Wn");
    scanf("%d %d", &year, &month);

    date = countDate(birth_year, birth_month, birth_day, year, month);
    printf("Wn%d Y %d M %d D and %d Y %d M 1 D has %d days.WnWn", birth_year, birth_month, birth_day,
    year, month, date + 1);

    printBiorhythm(date, year, month);
}

```

### 5.3. Result (snapshot)

Input a birthdate (ex: 1983 5 21)  
1985 7 2  
Insert wanted year and month. (ex: 2004 11)  
2006 1  
totaldays: 29

1985 Y 7 M 2 D and 2006 Y 1 M 1 D has 7488 days.

Biorhythm Result:

Low(-), High(\*), Danger(D)

1M 1D P:	12 D	M:	11 +
1M 2D P:	13 -	M:	12 +
1M 3D P:	14 -	M:	13 D
1M 4D P:	15 -	M:	14 D
1M 5D P:	16 -	M:	15 -
1M 6D P:	17 -	M:	16 -
1M 7D P:	18 -	M:	17 -
1M 8D P:	19 -	M:	18 -
1M 9D P:	20 -	M:	19 -
1M 10D P:	21 -	M:	20 -
1M 11D P:	22 D	M:	21 -
1M 12D P:	0 D	M:	22 -
1M 13D P:	1 +	M:	23 -
1M 14D P:	2 +	M:	24 -
1M 15D P:	3 +	M:	25 -
1M 16D P:	4 +	M:	26 -
1M 17D P:	5 +	M:	27 D
1M 18D P:	6 +	M:	0 D
1M 19D P:	7 +	M:	1 +
1M 20D P:	8 +	M:	2 +
1M 21D P:	9 +	M:	3 +
1M 22D P:	10 +	M:	4 +
1M 23D P:	11 D	M:	5 +
1M 24D P:	12 D	M:	6 +
1M 25D P:	13 -	M:	7 +
1M 26D P:	14 -	M:	8 +
1M 27D P:	15 -	M:	9 +
1M 28D P:	16 -	M:	10 +
1M 29D P:	17 -	M:	11 +
1M 30D P:	18 -	M:	12 +
1M 31D P:	19 -	M:	13 D

Press any key to continue . . .

Input a birthdate (ex: 1983 5 21)  
1991 11 21  
Insert wanted year and month. (ex: 2004 11)  
2016 5  
totaldays: 9

1991 Y 11 M 21 D and 2016 Y 5 M 1 D has 8807 days.

Biorhythm Result:

Low(-), High(\*), Danger(D)

5M 1D P:	20 -	M:	14 D	I:	28 -
5M 2D P:	21 -	M:	15 -	I:	29 -
5M 3D P:	22 D	M:	16 -	I:	30 -
5M 4D P:	0 D	M:	17 -	I:	31 -
5M 5D P:	1 +	M:	18 -	I:	32 D
5M 6D P:	2 +	M:	19 -	I:	0 D
5M 7D P:	3 +	M:	20 -	I:	1 +
5M 8D P:	4 +	M:	21 -	I:	2 +
5M 9D P:	5 +	M:	22 -	I:	3 +
5M 10D P:	6 +	M:	23 -	I:	4 +
5M 11D P:	7 +	M:	24 -	I:	5 +
5M 12D P:	8 +	M:	25 -	I:	6 +
5M 13D P:	9 +	M:	26 -	I:	7 +
5M 14D P:	10 +	M:	27 D	I:	8 +
5M 15D P:	11 D	M:	0 D	I:	9 +
5M 16D P:	12 D	M:	1 +	I:	10 +
5M 17D P:	13 -	M:	2 +	I:	11 +
5M 18D P:	14 -	M:	3 +	I:	12 +
5M 19D P:	15 -	M:	4 +	I:	13 +
5M 20D P:	16 -	M:	5 +	I:	14 +
5M 21D P:	17 -	M:	6 +	I:	15 +
5M 22D P:	18 -	M:	7 +	I:	16 D
5M 23D P:	19 -	M:	8 +	I:	17 D
5M 24D P:	20 -	M:	9 +	I:	18 -
5M 25D P:	21 -	M:	10 +	I:	19 -
5M 26D P:	22 D	M:	11 +	I:	20 -
5M 27D P:	0 D	M:	12 +	I:	21 -
5M 28D P:	1 +	M:	13 D	I:	22 -
5M 29D P:	2 +	M:	14 D	I:	23 -
5M 30D P:	3 +	M:	15 -	I:	24 -
5M 31D P:	4 +	M:	16 -	I:	25 -

Press any key to continue . . .

Input a birthdate (ex: 1983 5 21)  
1992 01 02  
Insert wanted year and month. (ex: 2004 11)  
2017 1  
totaldays: 29

1992 Y 1 M 2 D and 2017 Y 1 M 1 D has 9131 days.

Biorhythm Result:

Low(-), High(\*), Danger(D)

1M 1D P:	22 D	M:	2 +
1M 2D P:	0 D	M:	3 +
1M 3D P:	1 +	M:	4 +
1M 4D P:	2 +	M:	5 +
1M 5D P:	3 +	M:	6 +
1M 6D P:	4 +	M:	7 +
1M 7D P:	5 +	M:	8 +
1M 8D P:	6 +	M:	9 +
1M 9D P:	7 +	M:	10 +
1M 10D P:	8 +	M:	11 +
1M 11D P:	9 +	M:	12 +
1M 12D P:	10 +	M:	13 D
1M 13D P:	11 D	M:	14 D
1M 14D P:	12 D	M:	15 -
1M 15D P:	13 -	M:	16 -
1M 16D P:	14 -	M:	17 -
1M 17D P:	15 -	M:	18 -
1M 18D P:	16 -	M:	19 -
1M 19D P:	17 -	M:	20 -
1M 20D P:	18 -	M:	21 -
1M 21D P:	19 -	M:	22 -
1M 22D P:	20 -	M:	23 -
1M 23D P:	21 -	M:	24 -
1M 24D P:	22 D	M:	25 -
1M 25D P:	0 D	M:	26 -
1M 26D P:	1 +	M:	27 D
1M 27D P:	2 +	M:	0 D
1M 28D P:	3 +	M:	1 +
1M 29D P:	4 +	M:	2 +
1M 30D P:	5 +	M:	3 +
1M 31D P:	6 +	M:	4 +

Press any key to continue . . .

Input a birthdate (ex: 1983 5 21)  
1 1 1  
Insert wanted year and month. (ex: 2004 11)  
2014 2

1 Y 1 M 1 D and 2014 Y 2 M 1 D has 735233 days.

Biorhythm Result:

Low(-), High(\*), Danger(D)

2M 1D P:	14 -	M:	8 +	I:	25 -
2M 2D P:	15 -	M:	9 +	I:	26 -
2M 3D P:	16 -	M:	10 +	I:	27 -
2M 4D P:	17 -	M:	11 +	I:	28 -
2M 5D P:	18 -	M:	12 +	I:	29 -
2M 6D P:	19 -	M:	13 D	I:	30 -
2M 7D P:	20 -	M:	14 D	I:	31 -
2M 8D P:	21 -	M:	15 -	I:	32 D
2M 9D P:	22 D	M:	16 -	I:	0 D
2M 10D P:	0 D	M:	17 -	I:	1 +
2M 11D P:	1 +	M:	18 -	I:	2 +
2M 12D P:	2 +	M:	19 -	I:	3 +
2M 13D P:	3 +	M:	20 -	I:	4 +
2M 14D P:	4 +	M:	21 -	I:	5 +
2M 15D P:	5 +	M:	22 -	I:	6 +
2M 16D P:	6 +	M:	23 -	I:	7 +
2M 17D P:	7 +	M:	24 -	I:	8 +
2M 18D P:	8 +	M:	25 -	I:	9 +
2M 19D P:	9 +	M:	26 -	I:	10 +
2M 20D P:	10 +	M:	27 D	I:	11 +
2M 21D P:	11 D	M:	0 D	I:	12 +
2M 22D P:	12 D	M:	1 +	I:	13 +
2M 23D P:	13 -	M:	2 +	I:	14 +
2M 24D P:	14 -	M:	3 +	I:	15 +
2M 25D P:	15 -	M:	4 +	I:	16 D
2M 26D P:	16 -	M:	5 +	I:	17 D
2M 27D P:	17 -	M:	6 +	I:	18 -
2M 28D P:	18 -	M:	7 +	I:	19 -

Press any key to continue . . .

Input a birthdate (ex: 1983 5 21)

1234 1 2

Insert wanted year and month. (ex: 2004 11)

3211 1

1234 Y 1 M 2 D and 3211 Y 1 M 1 D has 722083 days.

Biorhythm Result:

Low(-), High(\*), Danger(D)

1M	1D	P: 20 -	M: 18 -	I: 9 +
1M	2D	P: 21 -	M: 19 -	I: 10 +
1M	3D	P: 22 D	M: 20 -	I: 11 +
1M	4D	P: 0 D	M: 21 -	I: 12 +
1M	5D	P: 1 +	M: 22 -	I: 13 +
1M	6D	P: 2 +	M: 23 -	I: 14 +
1M	7D	P: 3 +	M: 24 -	I: 15 +
1M	8D	P: 4 +	M: 25 -	I: 16 D
1M	9D	P: 5 +	M: 26 -	I: 17 D
1M	10D	P: 6 +	M: 27 D	I: 18 -
1M	11D	P: 7 +	M: 0 D	I: 19 -
1M	12D	P: 8 +	M: 1 +	I: 20 -
1M	13D	P: 9 +	M: 2 +	I: 21 -
1M	14D	P: 10 +	M: 3 +	I: 22 -
1M	15D	P: 11 D	M: 4 +	I: 23 -
1M	16D	P: 12 D	M: 5 +	I: 24 -
1M	17D	P: 13 -	M: 6 +	I: 25 -
1M	18D	P: 14 -	M: 7 +	I: 26 -
1M	19D	P: 15 -	M: 8 +	I: 27 -
1M	20D	P: 16 -	M: 9 +	I: 28 -
1M	21D	P: 17 -	M: 10 +	I: 29 -
1M	22D	P: 18 -	M: 11 +	I: 30 -
1M	23D	P: 19 -	M: 12 +	I: 31 -
1M	24D	P: 20 -	M: 13 D	I: 32 D
1M	25D	P: 21 -	M: 14 D	I: 0 D
1M	26D	P: 22 D	M: 15 -	I: 1 +
1M	27D	P: 0 D	M: 16 -	I: 2 +
1M	28D	P: 1 +	M: 17 -	I: 3 +
1M	29D	P: 2 +	M: 18 -	I: 4 +
1M	30D	P: 3 +	M: 19 -	I: 5 +
1M	31D	P: 4 +	M: 20 -	I: 6 +

Press any key to continue . . .