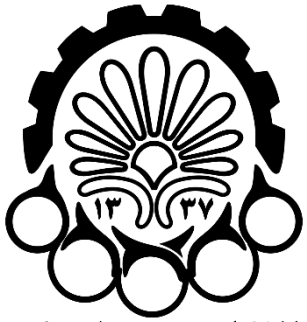
"به نام دادار دادگستر"



دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

کارگاه مبانی کامپیوتر و برنامه نویسی

عنوان

تمرین هفتم (C-Lab_loops)

مدرس

مهندس امیرحسین بابائیان

دانشجو

محمد یاراحمدی

۴۰۲۳۱۰۵۹

ترم پاییز ۰۳ - ۰۲
دانشکده مهندسی کامپیوتر، دانشگاه صنعتی امیرکبیر (پلی تکنیک تهران)

فهرست

پرسش ۱

اگر اعدادی که تنها از عوامل اول ۲ ، ۳ و ۵ تشکیل شده اند را اعداد زشت بدانیم، می خواهیم برنامه ای بنویسیم که در تشخیص زشت بودن یا نبودن یک عدد به ما کمک کند .

پاسخ پرسش ۱

برای حل این مسئله ، باید مقسوم علیه های عدد ورودی n را بیابیم . برای این امر اگر باقی مانده تقسیم عدد n بر یکی از اعداد بازه (۲,n/۲) صفر شد ، آن عدد مقسوم علیه n می باشد . برای آسودگی و کوتاه شدن خطوط کد تعیین مقسوم علیه ، میتوانیم از دستور حلقه بهره ببریم . اگر مقسوم علیه ای غیر از ۲ و ۳ و ۵ یافتیم عدد n زیباست در غیر اینصورت طبق تعریف این عدد زشت است .

```c
#include <stdio.h>
int main (void){
    int n ;
    scanf("%d" , &n);

    for(int i = 2 ; i <= n/2 ; i++){
        if((n % i == 0) && i != 2 && i != 3 && i != 5)
        {
            printf("Adad ziba ast.");
            return 1 ;
        }
        else {
            printf("Adad zesht ast.");
            return 1 ;
        }
    }
    return 0 ;
}
```

پرسش ۱ (قسمت دوم)

تمام عددهایی که ارقامشان فقط از دو رقم ۴ و ۷ تشکیل شده باشد را خوش شانس مینامیم. مثلا عدد ۴۷ ، ۷۴۴ یا ۴ به نظر او اعداد خو ش شانسی بوده اند که تمام رقم های شان ۴ و ۷ است اما عدد ۱۷ ، ۴۶۷ یا ۶ این شانس راندانشته اند. اعدادی همسایه اعداد خو ش شانس می نامیم که جمع تعداد ۴ و ۷ های آن ها یک عدد خوش شانس باشد . بنابراین، باید برنامه ای نوشته شود که با دریافت ورودی n تشخیص دهد که آیا این عدد یک همسایه برای اعداد خو ش شانس هست یا خیر...

پاسخ پرسش ۱ (قسمت دوم)

کاربر عدد ورودی را به صورت رقم به رقم وارد می کند و ما میتوانیم هر یک از این ارقام را به شکل کاراکتر آن عدد ورودی بگیریم . پس از اینکه کاربر هر رقم را وارد کرد ، خوش شانس بودن آن رقم بررسی می گردد و در صورت برقراری این شرط به تعداد شمارنده افزوده میگردد و این امر تا زمانی که کاربر در ورودی دادن به ترمینال از enter استفاده نکند ، تکرار خواهد شد . شرط پایان حلقه را کاراکتر enter قرار داده ایم .در صورت همسایه خوش شانس بودن شمارنده ، همین عبارت در خروجی چاپ خواهد شد .

```c
#include <stdio.h>
int main (){

    char c ;
    int count = 0 ;
    while ( c != '\n')
    {
        scanf("%c", &c);
        if ( c == '4' || c == '7')
            count ++ ;
    }
    if ( count == 4 || count == 7 || count == 47 || count == 77 || count == 44 ||
count == 74)
    printf("lucky number neighbor");
    else
    printf("unlucky number neighbor");

    return 0 ;
}
```

پرسش ۲

برنامه ای بنویسید که با دریافت عدد n از کاربر، مقدار جمع زیر را یکبار از راست به چپ و یکبار از چپ به راست محاسبه کند:

$$f(n) = ۱ + \frac{۱}{۲} + \frac{۱}{۳} + \frac{۱}{۴} + \cdots + + \frac{۱}{۲^n}$$

پاسخ پرسش ۲

با استفاده از هدرفایل math.h شرط پایان حلقه را تعیین می کنیم . طبق خواسته سوال ، حاصل عبارت داده شده را یکبار از چپ به راست و یکبار از راست به چپ محاسبه میکنیم . جواب بدست آمده با خاصیت جا به جایی جمع در ریاضیات مطابقت ندارد .

دلیل این اختلاف در ورودی های بزرگ این است که کامپیوتر اعداد را بصورت نماد علمی ذخیره می کند و از تقریب بهره میبرد لذا هر چه ورودی بزرگتری بدهیم حاصل $\frac{۱}{۲^n}$ را با تقریب بیشتری محاسبه می کند و در برابر اعداد بزرگتر عبارت f(n) مثل ۱ ، با از آن چشم پوشی می کند .

```c
#include <stdio.h>
#include <math.h>
int main (void){

    int n ;
    scanf("%d" , &n);
    double sum1 = 0 ;
    int g = pow(2 , n);
    for (int i = 1 ; i <= g ; i++){
        sum1 += (1.0/i) ;
    }
    printf("%lf\n",sum1);

    double sum2= 0 ;
    for (int i = g ; i > 0 ; i--){
        sum2 += (1.0/i) ;
    }
    printf("%lf",sum2);
    return 0 ;
}
```

پرسش ۳

برنامه ای بنویسید که دو الگو زیر که به ازای n = ۵ داده شده را به ازای هر ورودی n در ترمینال چاپ کند .

A	۱ ۱
BB	۲۱ ۱۲
CCC	۳۲۱ ۱۲۳
DDDD	۴۳۲۱ ۱۲۳۴
EEEEE	۱۲۳۴۵۴۳۲۱
الگو ۱	الگو ۲

پاسخ پرسش ۳

در الگو ۱ ، تعداد کاراکتر مجاز هر سطر با توجه به شماره آن سطر تعیین می شود . کاراکتر آغازی 'A' را برای سطر اول در نظر میگیریم و با توجه به شماره سطر اول که ۱ است ، این کاراکتر را یکبار چاپ می کنیم و به خط بعدی میرویم . حالا در سطر بعدی علاوه بر رعایت شرط قبل ، کاراکتر هم باید آپدیت شود که اینکار با استفاده از جمع "اسکی کد" کاراکتر 'A' با شماره سطر قبل صورت می گیرد .

در الگو ۲ ، در سطر iام باید از عدد ۱ تا i چاپ شود و بعد به اندازه کافی (در خط ۱۸ هم کد ، نحوه محاسبه تعداد کاراکتر فاصله ها قرار داده شده است) کاراکتر قاصله چاپ گردد . حال باید در انتهای خطوط ، اعداد را از عدد i تا ۱ چاپ کنیم .با این حرکت در خط آخر کد عدد ورودی n ۲ مرتبه چاپ خواهد شد که با قرار دادن یک شرط ، جلوی این اتفاق را خواهیم گرفت .

```c
#include<stdio.h>
int main(){
    int n;
    scanf("%d" , &n);
    for(int i=0 ; i<n ; i++){
        for(int j=0 ; j<=i ; j++){
            printf("%c" , 'A' + i);
        }
        printf("\n");
    }
    printf("\n");

    int spaces  ;
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= i; j++) {
            printf("%d", j);
        }
        spaces = 2 * (n - i) - 1;
        for (int j = 1; j <= spaces; j++) {
            printf(" ");
        }
        for (int j = i; j >= 1; j--) {
            if (j == n ) continue;
            else    printf("%d", j);
        }
        printf("\n");
    }
    return 0 ;
}
```