به نام خدا

دستور کار کارگاه مبانی کامپیوتر و برنامه‌نویسی

جلسه دهم

توابع بازگشتی و برنامه‌نویسی پویا

۱. رابطه‌ی بازگشتی دنباله‌ی فیبوناچی به شکل زیر می‌باشد:

1 1 2 3 5 …

$$f(n) = f(n-1) + f(n-2)$$

این رابطه را می‌توان به صورت زیر در زبان C پیاده‌سازی کرد:

```c
#include <stdio.h>

int fibonacci(int n) {
    if (n == 1 || n == 2) {
        return 1;
    }
    return fibonacci(n - 1) + fibonacci(n - 2);
}

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

}
```

پیاده‌سازی فوق را به گونه‌ای تغییر دهید که عدد n را از کاربر گرفته و جمله‌ی nام فیبوناچی را چاپ کند. اگر n برابر ۱۰۰ باشد برنامه‌ی شما می‌تواند جواب را در چند دقیقه تولید کند؟

همانطور که مشاهده کردید، جواب در چند دقیقه تولید نمی‌گردد، با توجه به اینکه از نظر منطقی برای محاسبه جمله‌ی ۱۰۰ام حداکثر نیاز به محاسبه‌ی ۱۰۰ جمله‌ی قبل می‌باشد چرا این محاسبه به این میزان طولانی شده است؟

برای اینکه دلیل این موضوع را بهتر متوجه شوید برنامه‌ی فوق را به گونه‌ای تغییر دهید که هر بار فراخوانی تابع فیبوناچی را با پارامتر آن نمایش دهد. خروجی‌ها را برای nهای متفاوت بررسی کنید. آنچه می‌بینید با مدرس خود در میان بگذارید. آیا دلیل این طولانی شدن را پیدا کرده‌اید؟ ارتباطی بین این سوال و قسمت امتیازی سوال سوم تمرین پنجم خود می‌بینید؟

۲. تابع فیبوناچی را به صورت غیربازگشتی بنویسید، این بار تابع را با n برابر ۱۰۰ فراخوانی کنید، آیا جواب حاصل در زمان چند دقیقه حاصل می‌گردد؟

اینطور به نظر می‌رسد که راه‌حل‌های بازگشتی در برخی از موارد کارآیی لازم را ندارند، به نظر شما آیا روشی برای بهبود این موضوع وجود دارد یا واقعا استفاده از روش‌های بازگشتی بی‌ثمر است؟

۳. یکی از روش‌های بهبود زمان اجرا در توابع بازگشتی استفاده از روش memorization می‌باشد. در ادامه کد سی تابع فیبوناچی را که با این روش بهبود یافته است می‌بینید:

```c
#include <stdio.h>

// stores fibonacci sequences in the array.
// global variables are initialized to zero in c
int memory[1000];

int fibonacci(int n) {
    if (n == 1 || n == 2) {
        return 1;
    }
    if (memory[n] == 0) {
        memory[n] = fibonacci(n - 1) + fibonacci(n - 2); // memorizes what it
calculated now.
    }
    return memory[n];
}
```

به نظر شما این روش چگونه سرعت محاسبه را افزایش می‌دهد؟ آن را برای مدرس خود تشریح کنید.

۴. تابع بازگشتی زیر را یکبار به روش معمول و بار دیگر به روش دیگر memorization پیاده‌سازی کنید.

$$f(x) = f(x-1) + f(x-2) + 2f(x-3)$$

یکی دیگر از روش‌های افزایش سرعت در محاسبات بازگشتی روش برنامه‌نویسی پویا می‌باشد.

۵. این روش برنامه‌نویسی پویا به جای فراخوانی تابع به صورت بازگشتی از یک آرایه استفاده می‌کنیم و به این ترتیب مقدارهای قبلی در آرایه ذخیره می‌شوند و می‌توان از آن‌ها به سادگی استفاده کرد.

در زیر پیاده‌سازی برنامه‌ی معروف فیبوناچی با استفاده از برنامه‌نویسی پویا را می‌بینید:

```c
#include <stdio.h>

int main(int argc, const char *argv[]) {
    int n;
    scanf("%d", &n);

    int D[n];

    D[0] = 1; // f(0) = 1, fibonacci sequence number is 1
    D[1] = 1; // f(1) = 1, fibonnaci sequence number is 2
    for (int i = 2; i < n; i++) {
        D[i] = D[i - 1] + D[i - 2];
    }
    printf("%d\n", D[n - 1]);
}
```

چگونگی کارکرد آن را با مدرس خود در میان بگذارید. برنامه را با اعداد بزرگ بررسی کنید، آیا زمان پاسخ‌دهی معقول است؟

در دوران دبیرستان مفهوم ترکیب آشنا شدید که رابطه‌ی بازگشتی آن به شکل زیر است:

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$$

$$\binom{n}{0} = 1$$

$$\binom{n}{1} = n$$

این رابطه را با استفاده از برنامه‌نویسی پویا پیاده‌سازی کنید

۶. برنامه‌ای بنویسید که رشته‌ای را از ورودی خوانده، مشخص کند که آیا رشته از هر دو طرف که در نظر گرفته شود یکسان است یا خیر (مثلاً رشته‌ی "beeb" چنین خاصیتی دارد.). در نظر داشته باشید که رشته می‌تواند شامل کاراکترهای فاصله و ... هم باشد.

```
A man A plan A canal panama
True
Ah Satan sees Natasha
True
```