جمع موازی اعداد

مقدمه

در این تمرین قصد داریم برنامه‌ای بنویسیم تا به کمک آن بتوان جمع اعداد ۱ تا `n` را به صورت موازی و در چند فرآیند حساب کرد. بدین ترتیب برنامه‌ی ما عدد `n` و تعداد فرآیندها (`m`) را از ورودی می‌گیرد و جمع اعداد ۱ تا `n` را در `m` فرآیند حساب می‌کند.

شرح سازوکار برنامه

برنامه باید بازه‌ی گفته شده از اعداد را به `m` قسمت تقسیم کند و جمع اعداد هر بازه را به یک فرآیند جدا بسپارد؛ بدیهی است که در این حالت به تعداد `m` فرآیند خواهیم داشت و همچنین امکان دارد بازه‌ی پایانی اعداد کمتری را در خود جای دهد.

برای مثال اگر `n = 10` و `m = 3` باشد آنگاه ما سه بازه خواهیم داشت که دو بازه‌ی ابتدایی به ترتیب اعداد از ۱ تا ۴ و از ۵ تا ۸ را شامل خواهند شد و بازه‌ی انتهایی اعداد ۹ و ۱۰ را در خود خواهد داشت.

از شما خواسته می‌شود برنامه‌ای بنویسید که با گرفتن اعداد `n` و `m` حاصل جمع اعداد ۱ تا `n` را توسط `m` رشته چاپ کند.

اعداد `n` و `m` به شکل زیر خواهند بود:

$$1 \leq n \leq 100000$$

$$2 \leq m \leq 100$$

نمونه‌ی ورودی و خروجی برنامه

ورودی ۱:

```
100
4
```

خروجی ا:

```
5050
```

نکته‌های مهم در پیاده‌سازی

- **شما تنها امکان استفاده از توابع تدریس شده در کلاس و اسلایدها را خواهید داشت**
- **شما تنها امکان استفاده از سرآیندهای تدریس شده را خواهید داشت**
- **ضمانت می‌شود برنامه به صورت کامل توسط توابع تدریس شده قابل پیاده‌سازی است**
- **برنامه‌ی شما توسط کوئرا نمره‌دهی و داوری نخواهد شد و توسط نرم‌افزار داوری خصوصی بر روی سیستم‌عامل لینوکس کامپایل و اجرا خواهد شد**

ضرب ماتریس‌ها

مقدمه

در این تمرین از شما خواسته می‌شود که تابع مربوط به ضرب دو ماتریس را در برنامه‌ی زیر کامل کنید:

```
#include <stdio.h>
#include <stdlib.h>

// PUT YOUR CODE IN THIS FUNCTION
void matrixMultiply(int** matrixA, int rowsA, int colsA,
                    int** matrixB, int rowsB, int colsB,
                    int** result)
{
    if (colsA != rowsB)
    {
        printf("Error: Incompatible matrix dimensions for multiplicat
        return;
    }

    // PUT YOUR CODE HERE
}

void printMatrix(int** matrix, int rows, int cols)
{
    for (int i = 0; i < rows; i++)
    {
        for (int j = 0; j < cols; j++)
        {
            printf("%d\t", matrix[i][j]);
        }
        printf("\n");
    }
}

int main()
{
    int rowsA, colsA, rowsB, colsB;

    scanf("%d", &rowsA);
```

```
scanf("%d", &colsA);
scanf("%d", &rowsB);
scanf("%d", &colsB);

int** matrixA = malloc(rowsA * sizeof(int*));
int** matrixB = malloc(rowsB * sizeof(int*));
int** result = malloc(rowsA * sizeof(int*));

for (int i = 0; i < rowsA; i++)
{
    matrixA[i] = (int*)malloc(colsA * sizeof(int));
}

for (int i = 0; i < rowsB; i++)
{
    matrixB[i] = (int*)malloc(colsB * sizeof(int));
}

for (int i = 0; i < rowsA; i++)
{
    result[i] = (int*)malloc(colsB * sizeof(int));
}

for (int i = 0; i < rowsA; i++)
{
    for (int j = 0; j < colsA; j++)
    {
        scanf("%d", &matrixA[i][j]);
    }
}

for (int i = 0; i < rowsB; i++)
{
    for (int j = 0; j < colsB; j++)
    {
        scanf("%d", &matrixB[i][j]);
    }
}

matrixMultiply(matrixA, rowsA, colsA, matrixB, rowsB, colsB, resu

if (colsA == rowsB)
{
```

```
        printMatrix(result, rowsA, colsB);
    }

    for (int i = 0; i < rowsA; i++)
    {
        free(matrixA[i]);
        free(result[i]);
    }
    for (int i = 0; i < rowsB; i++)
    {
        free(matrixB[i]);
    }
    free(matrixA);
    free(matrixB);
    free(result);

    return 0;
}
```

کد بالا دو ماتریس را از ورودی خوانده و توسط تابع `matrixMultiply` آنها را در هم ضرب کرده و در ماتریس `result` قرار می‌دهد.

شرح سازوکار برنامه

برنامه‌ی شما باید با کمک نخ‌ها عملیات ضرب را انجام دهد. ترتیب نوشتن تابع بدین صورت است که عملیات ضرب هر سطر ماتریس `A` در هر ستون ماتریس `B` باید در یک نخ انجام شود؛ به عبارتی برای یک ماتریس ۳ در ۳ در ماتریس ۳ در ۲ به ۶ نخ نیازمندیم.

ورودی و خروجی برنامه

نمونه ورودی

در ابتدا ۴ عدد که به ترتیب بیانگر تعداد سطر و ستون ماتریس `A` و تعداد سطر و ستون ماتریس `B` هستند خواهند آمد و پس از آن ماتریس‌های `A` و `B` . به مثال زیر توجه کنید:

```
3
3
3
```

```
2
3 3 3
3 3 3
3 3 3
1 1
1 1
1 1
```

نمونه خروجی

برنامه‌ی شما تنها باید ماتریس حاصل را در خروجی بدون هیچ اضافاتی چاپ کند. به مثال زیر توجه کنید:

```
9 9
9 9
9 9
```

پیداکردن کلمه در جمله

مقدمه

در این سوال قصد داریم تا برنامه‌ای بنویسیم که رشته‌ای را به وسیله‌ی حافظه‌ی اشتراکی از برنامه‌ای دیگر دریافت و تعداد واژه‌های `BSD` و `Linux` را در آن بشمارد.

شرح سازوکار برنامه

برنامه‌ی شما به دو حافظه‌ی اشتراکی با کلیدهای `11021` و `11022` متصل خواهد شد. سپس یک رشته به طول `2048` کاراکتر را از حافظه‌ی نخست با کلید `11021` خوانده و پس از آن دو نخ مجزا ساخته و در هرکدام شروع به جستجو برای یافتن تعداد یکی از واژگان بالا می‌کند. پس از اتمام جستجو تعداد واژگان را با هم جمع کرده و به عنوان `int` در حافظه با کلید `11022` قرار می‌دهد.

خروجی‌ها و ورودی‌های برنامه

*برنامه هیچ ورودی و خروجی نخواهد داشت و تنها از راه حافظه اشتراکی ارتباط برقرار می‌کند. *

نمونه رشته‌ی ورودی

```
"Exploring the differences between BSD and Linux can be fascinating.
```

نمونه‌ی عدد نهایی

```
4
```