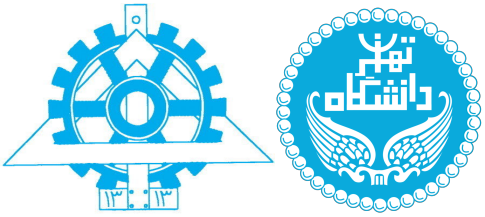
به نام خدا



برنامه‌نویسی موازی - پاییز 1403
تمرین کامپیوتری ۲ - موازی سازی در نخ (با OpenMP)

مقدمه

همانطور که می‌دانید، امروزه نیازمند حجم بسیار زیادی از محاسبات هستیم؛ از شبیه‌سازی‌های علمی گرفته تا تحلیل‌های لحظه‌ای. برخی از این محاسبات ممکن است بسیار پیچیده بوده و یا نیاز به ارائه نتایج در زمانی محدود باشند. برای رسیدن به این نیازها، اغلب باید برنامه‌های خود را سریع‌تر کنیم. یک راه حل، اجرای موازی وظایف، به کمک رابطِ برنامه‌نویسیِ کاربردیِ[1] OpenMP است. در این تکلیف، شما کاربرد OpenMP در افزایش سرعت کدهایی که محاسبات متنوعی انجام می‌دهند را مشاهده خواهید کرد.

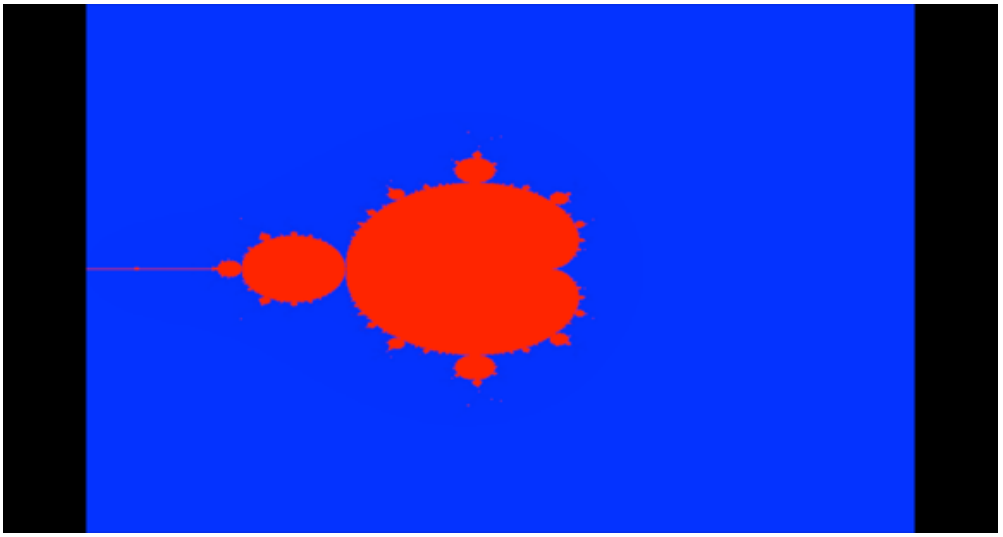
شرح تمرین

در **مسئله اول**، شما با نظریهٔ Mandelbrot Set کار خواهید کرد که به مفهوم واگرایی می‌پردازد. این نظریه بررسی می‌کند که چگونه تکرارِ مربعی کردنِ یک عدد و افزودن یک ثابت می‌تواند منجر به این شود که نتیجه، در شرایط خاص به بی‌نهایت میل نکند و در نتیجه همگرا شود.

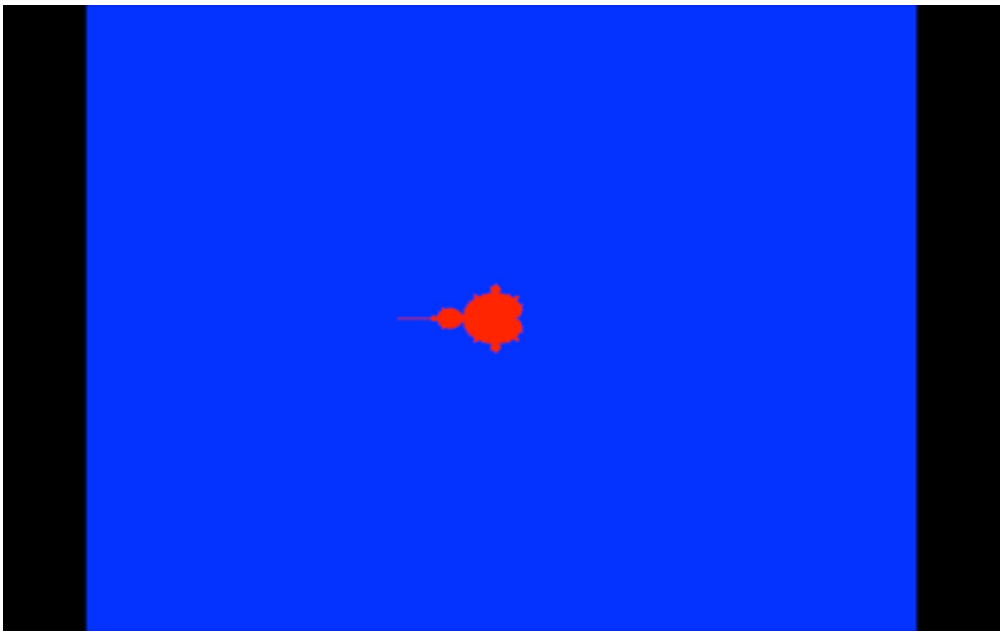
هدف از این تمرین به تصویر کشیدن این همگرایی در اعداد مختلط است. در این نظریه که فرمول اصلی آن به شکل $z = z^2 + c$ می‌باشد، اگر $z = 0$ باشد، با اشکال کلاسیک مندلبرو مواجه خواهیم شد و با تغییر c میتوانیم در شکل فراکتال[2] zoom in یا zoom out کرده و اشکالی پیچیده اما منظم را مشاهده کنیم.
وظیفهٔ شما این است که یک محاسبه‌گر و شبیه‌ساز Mandelbrot Set را پیاده‌سازی کنید و سپس با استفاده از OpenMP آن را موازی کنید.

در این مسئله شما باید الگوریتم مسئله را مرتبا اعمال کرده و مشاهده کنید که در صورت تغییر عدد ثابت، شکل فراکتال ثابت است اما zoom in و zoom out اتفاق می‌افتد و در صورت عوض کردن z، شکل دیگری تشکیل می‌شود که می‌تواند همگرا هم نباشد.

[1] Application Programming Interface (API)
[2] Fractal



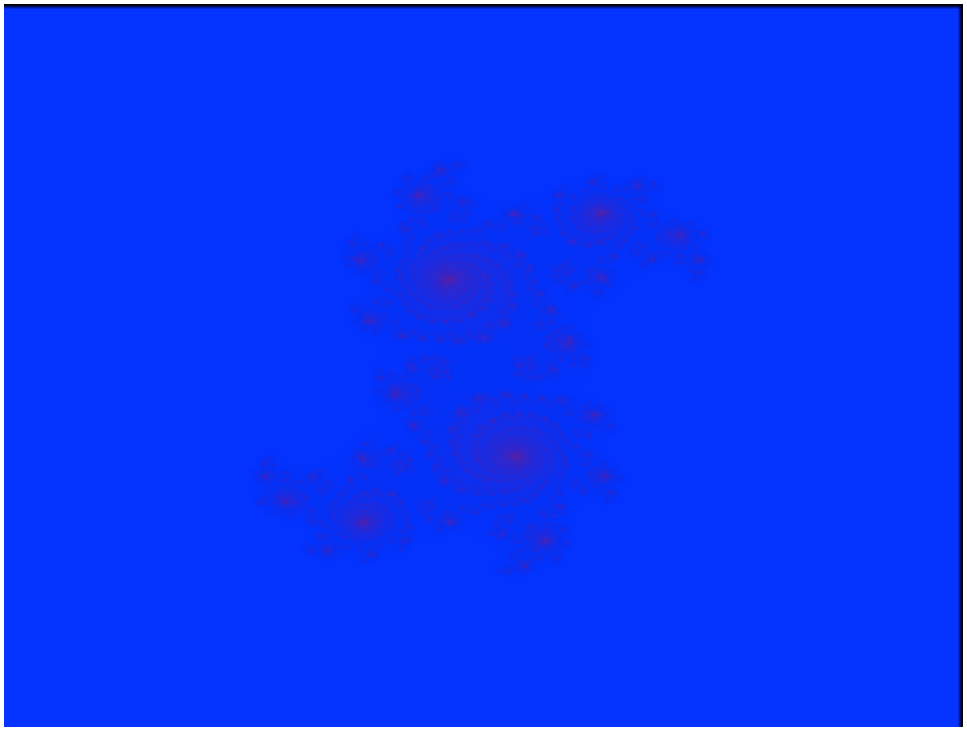
شکل ۱. نمونه خروجی شبیه‌ساز برای حالتی که z = 0 و c = 4j باشد.



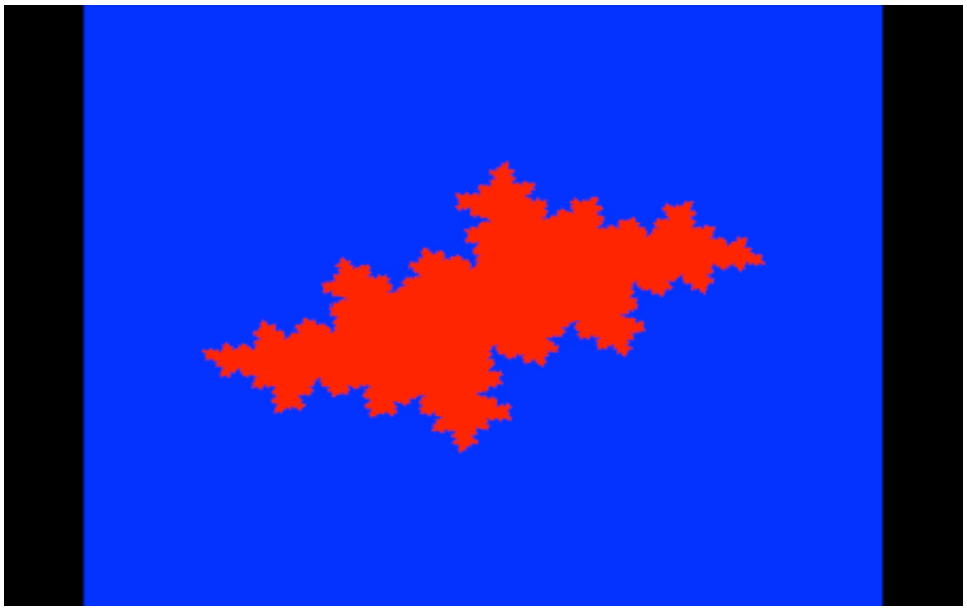
شکل ۲. نمونه خروجی شبیه‌ساز برای حالتی که z = 0 و c = 16j باشد.

در **مسئله دوم**، شما با Julia Set که ارتباط نزدیکی با Mandelbrot Set دارد و اصول ریاضی مشابهی را دنبال می‌کند، آشنا خواهید شد. فرآیند ایجاد Julia Set نیز شامل مربعی کردن اعداد مختلط و افزودن ثابت‌ها برای مشاهدهٔ الگوهای واگرایی است، اما با دو عدد ثابت برای هر نقطه. شما Julia Set را پیاده‌سازی خواهید کرد و از OpenMP برای بهبود کارایی آن استفاده می‌کنید.

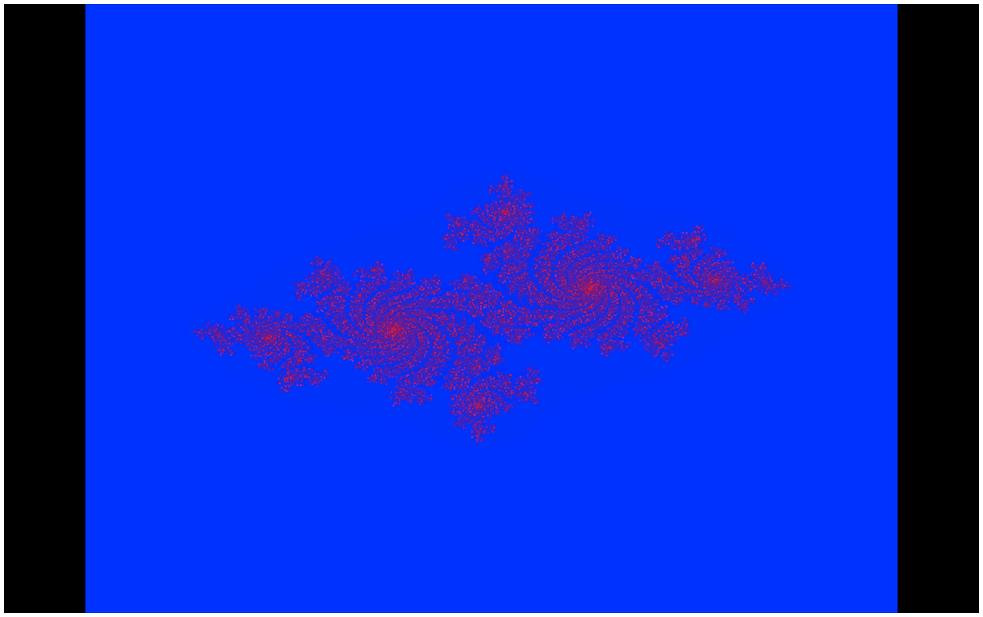
در این مسئله شما باید الگوریتم مسئله را مرتبا اعمال کنید و مشاهده که اشکال زیبایی به وجود می‌آیند.



شکل ۳. نمونه خروجی شبیه‌ساز برای حالتی که z = 0 و c=(0.355, 0.355j) باشد.



شکل ۴. نمونه خروجی شبیه‌ساز برای حالتی که z = 0 و c=(-0.5, 0.5j) باشد.



شکل ۵. نمونه خروجی شبیه‌ساز برای حالتی که z = 0 و c=(-0.7, 0.27015j) باشد.

برای درک بهتر مسئله‌های اول و دوم می‌توانید به این لینک مراجعه کنید.

توجه شود خواسته ما در مسئله اول و دوم پیاده‌سازی معادلهٔ مندلبرو و جولیا می‌باشد، همچنین لازم است تا این معادله را به اندازه کافی تکرار کنیم تا به مقادیر مشابه برسیم. به منظور صحت‌سنجی پیاده‌سازی خود، می‌توانید در نظر بگیرید در تمامی شکل‌های خروجی، تعداد iteration های الگوریتم 1000 بوده است. پیاده‌سازی شما می‌بایست به ازای تعداد تکرارهای مختلف، خروجی درستی تولید کند.

همچنین لازم است تا نتایج معادله در iteration ها را روی جدول مختصات اعداد مختلط و حقیقی مشاهده نمایید و سپس هر دو کار (محاسبه و شبیه‌سازی روی جدول مختصات) را به صورت موازی نیز تکرار نمایید.

شما ابتدا باید محاسبه‌گر را به تعداد کافی تکرار کنید و تمام نقاط ساخته شده توسط محاسبه‌گر را ذخیره و در شبیه‌ساز نقاط ساخته شده را در صفحه ی مختصات نمایش دهید.

مشاهده ویدیوی مربوط به مسئله‌های اول و دوم نیز به شدت به داشتن شهودی از مسئله کمک خواهد کرد.

در نهایت در **مسئله سوم** شما روش Monte Carlo را برای تخمین عدد π به کار خواهید برد. این روش شامل تولید نقاط تصادفی در یک مربع و محاسبه تعداد نقاطی است که در یک دایرهٔ واحد قرار می‌گیرند. نسبت نقاط داخل دایره به کل نقاط، تقریبی از π ارائه می‌دهد. شما این فرآیند را

شبیه‌سازی کرده و سپس با استفاده از OpenMP موازی‌سازی خواهید کرد تا تأثیر آن بر سرعت محاسبه را مشاهده کنید.

برای درک بهتر مسئله سوم می‌توانید به این لینک مراجعه کنید.

نکات و نحوه تحویل

- برای هر کدام از سوال‌ها نسبت زمان حالت سریال به حالت موازی را در قالب speedup گرفته شده در خروجی چاپ کرده و در گزارش خود توجیه کنید.
- توجه کنید که گرفتن speedup معقول بخشی از نمره تمرین شما را شامل می‌شود.
- در نظر داشته باشید، نسخهٔ سریال می‌بایست در حالت بهینه بوده و پیاده‌سازی نسخهٔ موازی با نسخهٔ سریال منطبق باشد. قسمتی از ارزیابی پیاده‌سازی شما به همین موضوع اختصاص پیدا خواهد کرد.
- کدهای شما می‌بایست به زبان C/C++ و در سیستم عامل Windows و یا Linux قابل کامپایل و اجرا باشند.
- حتما در ابتدای فایل برنامه ارسالی، نام، نام خانوادگی و شمارهٔ دانشجویی اعضای گروه ذکر شود.
- تنها یکی از اعضای گروه پاسخ تمرین را آپلود کند.
- در محل بارگذاری در سایت درس، فایل‌ها و کدهای مورد نیاز به همراه گزارش پروژه را بارگذاری نمایید.
- در صورت داشتن سوال می‌توانید از طریق ایمیل با طراحان تمرین در ارتباط باشید.
- تمامی مواردی که در گروه و فروم درس ذکر می‌شوند جزئی از تمرین خواهند بود.
- هدف این تمرین یادگیری شماست، لطفا تمرین را خودتان انجام دهید.
- در صورت محرز شدن تقلب، مطابق با سیاست‌های درس برخورد خواهد شد.
- طراحان: سپهر مدیرصانعی، سروش صادقیان

موفق باشید.