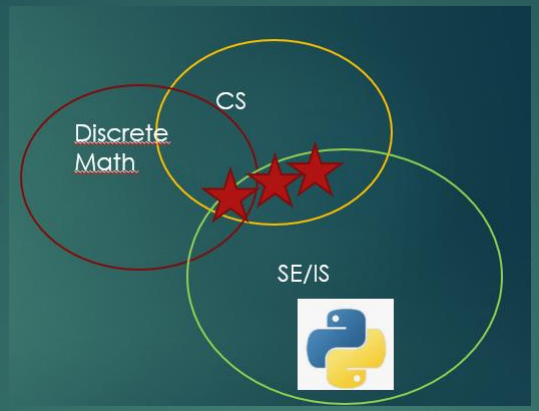

Especificación (SPEC) de la Tarea: Karatsuba Python

DR. CARLOS LORÍA-SÁENZ

LIBERADO: FEBRERO 2019

ESTRUCTURAS DISCRETAS EIF203 /UNA

Introducción

- El objetivo de este documento es especificar los alcances y entregables de una tarea programada sobre el algoritmo de Karatsuba (para multiplicación rápida) y su análisis de tiempo
- Acá se definen los resultados esperados, incluyendo la forma de implementación obligatoria
- Se establecen los mecanismos de organización, forma de entrega, revisión y evaluación

Resumen de productos

- Una aplicación en Python de consola que permita estudiar y verificar tiempos de corrida de Karatsuba comparados con algoritmos naturales ("*de la escuela*") de multiplicación
- Realizar una investigación sobre el de Karatsuba que permita su implementación y con ella generar un pequeño reporte sobre dicho algoritmo y su tiempo de corrida

Objetivos Generales

- Poner en práctica técnicas de análisis de algoritmos vistas en clase
- Ejercitar divide-y-conquista para solución recursiva de problemas
- Experimentar con algoritmia en Python durante el ejercicio
- Fomentar la investigación sobre temáticas relacionadas al curso
- Ejercitar trabajo en equipo

Tareas

- Investigar Karatsuba y emitir un reporte escrito sobre el mismo y con los resultados de su implementación
- Implementar artimética "natural" ("*de la escuela*") (se debe implementar suma, resta, multiplicación, división, comparación. Se debe manejar complemento a la base para la resta. Sin embargo, los casos de prueba serán siempre en base 10.
- Implementar Karatsuba por herencia siguiendo un modelo que será indicado por el profesor
- Permitir leer casos de prueba de disco que ejerciten los algoritmos de manera flexible
- Hacer casos de prueba para evaluar y mostrar resultados
- Mostrar apropiadamente los resultados para poder verificar correctitud
- Generar salidas de tiempo de corrida que comparen los algoritmos naturales
- Generar gráficas de tiempo y analizar con relación al análisis teórico en el reporte escrito

Consideraciones iniciales

- La tarea de implementación requiere nociones básicas de programación OOP en Python
- Se explicará en clase/consulta lo necesario para poder trabajar de esa manera
- Su implementación debe cumplir con el modelo OOP que se le pida en dichas clases
- Se estima que con unas 2 horas de explicación se puede trabajar en el proyecto. Parte de hará en una clase, parte en videos complementarios

Requerimientos de organización

- Debe hacerse en los grupos previamente inscritos según se pidió al inicio del semestre
- No se acepta de otra forma
- No se aceptan nuevos grupos luego de la fecha que fue indicada. SIN EXCEPCIÓN.

Requerimientos diseño e implementación

- Usar Python OOP para modelar números que permitan aritmética "natural" y el algoritmo de Karatsuba por herencia
- Separar en módulos (archivos .py) cada funcionalidad (no todo en un solo .py)
- Usar sobrecarga de operadores para mayor facilidad de uso (se explicará cómo)
- Leer casos de prueba de disco
- Generar salidas de corridas en formatos que permitan importación en Excel
- Documentar adecuadamente cada clase y método

Requerimientos del reporte de investigación

- Una parte del trabajo demanda investigar sobre el algoritmo Karatsuba
- Debe producirse un reporte sobre el algoritmo, la estrategia que lo define, ejemplos de uso, tiempo de corrida teórico (según literatura), resultados propios del estudiante sobre si se confirma ese tiempo usando su implementación
- El reporte debe tener una estructura formal: portada, índice de contenido, cuerpo con secciones bien definidas bibliografía, sin errores gramaticales, no "copy-paste" de otros. Máximo 10 páginas en total, espacio sencillo, Times New Roman, 11pts

Evaluación y Valor App

- Demo en clase: 60% (unos 12 minutos por grupo)
- Mejor traerlo en su propia laptop por si acaso haya problemas de setup en el lab. Si no logran la demo por razones ajenas al profesor reciben cero
- Revisión de código 40%. Se revisará que se implementa según lo solicitado
- Se puede otorgar hasta un 20% extra sobre la nota si se añaden más funcionalidades o herramientas (previamente aprobadas por el profesor). Sólo aplica si la aplicación tiene un mínimo obligatorio

Valor en la Nota

- Este trabajo aporta un valor equivalente a 5 quices (en rubro de actividades de aprendizaje de la carta al estudiante). La parte programada vale 60% el reporte 40% de la nota obtenida.
- Podrá según calidad aportar al rubro de extras a criterio del docente.
- Si hay extras deben ser aprobados previamente por el profesor y sólo aplican si se obtiene un mínimo en lo pedido obligatoriamente

Entregables

- Se subirá **puntualmente** el día y hora que será comunicado por el profesor a un sitio DROPBOX (dedicado para ese fin) por un miembro del grupo según será designado por el profesor (coordinador).
- El proyecto se sube en un `.zip` (o equivalente) llamado `Karatsuba_Python_AAA_GG`, donde `AAA` es el nombre del que sube el proyecto y `GG` es el `id` del grupo autor que el profesor le asignó.
- Dentro del `zip` debe venir un directorio (carpeta) `Karatsuba` (con el proyecto) y un `README.txt` con los nombres de los autores, su horario. Debe estar organizado en src (fuentes), data (salidas del programa), test (casos de prueba), doc (documentación)
- El reporte **debe ser en papel**, no digital. Sino se enterga el reporte no se revisa el proyecto del todo.
- Incumplimiento de estos requisitos anula el proyecto ni siquiera se revisa aunque funcione. Reciben cero. Sin derecho a reclamo.

Fecha y hora de entrega

- El día exacto de entrega se da a conocer oportunamente en clase y por los medios usuales en el curso.
- Ese día de entrega se sube el entregable en la forma requerida y se revisa la demo en la hora de matrícula.
- En la demo se usa exactamente la misma versión entregada al profesor. Cualquier disparidad anula el proyecto. La prueba no debe depender de que haya Internet disponible.
- Impuntualiddes por razones injustificadas podrán recibir pérdida de puntos de hasta 10 puntos por minuto de atraso hasta llegar a cero.
- Es obligación estar presente durante la demo. El ausente recibe cero completo en todo el trabajo.

Guía de revisión

- Se le entregará oportunamente una guía de revisión la cual deberá ser impresa y traída el día de la demo.
- Esta indicará cómo se revisará la demo.
- El profesor podrá usar casos de prueba para revisar/verificar el funcionamiento de su programa. **Nota**: Se adjuntan dos modelos de casos de prueba en el directorio src de este spec
- Cada grupo debe por su parte tener sus propios casos de prueba también que el profesor puede requerir ver

Posibles Extras

- Jupyter
- Pydoc (generar html)
- Git/Github
- Unittest