SAGE como calculadora avanzada

- Función() == objeto.metodo()
- Def f(paramtros):
- Variables simbólicas -> var ('nombres de variables')
- Sacar factores de una expresión: objeto.factor()
- Simplificar una expresión : objeto.simplify()
- Gráficas mirar los apuntes (meter ejemplo)
- Resolucion de ecuaciones: solve(ecuación)
- Limites: función.limit() o limit()
- Series: sum(expresion, variable, limite inferior, limite superior)
- Derivadas: función.derivative()
- Integrales: función.integrate()

Estructuras de datos: Listas

- Creacion: L=list(), L=[elemento,…,elemento]
- Longitud: len(L)
- Concatenar: L+K
- Copiar: L=list(K)
- Añadir elemento: L.append(elemento)
- Ordenar una lista: L.sort()
- Mirar srange específicos en la teoría
- Dar la vuelta a una lista: L.reverse()
- Sumar elementos: L.sum()
- Eliminar: L.remove()
- Acceder a un elemento: L[i] (mirar especiales)
- Digitos de un numero en una lista: L=numero.digits() (la da al reves)
- Factorizar números: numero.factor()

Estructuras de datos: Tuplas (no se puede asignar nuevos valores)

- Creacion: T=tuple(), T=(elemento,…,elemento)
- Se usan las primitivas de lista

Estructuras de datos: Conjuntos

- Creacion: A=set(), A={element}
- Añadir elementos: A.add()
- Eliminar: A.remove()
- Mismas primitivas que en lista

Técnicas de programación

- Bucles for: for elemento in contenedor: (srange(primero,ultimo+1))
- Bucles while: while condicion:
- Orbitas:
  def orbita(ini,f):

```
L = []
while not(ini in L):
        L.append(ini)
        ini = f(ini) #Actualiza el valor de ini
 return L
```

- Potencias: existe un algoritmo en las hojas

Teoría de números

- Mcd: gcd(a,b) (para bezout xgcd)
- Saber si es un numero primo: .is_prime()
- Siguiente primo: .next_prime()
- Lista de primos entre dos números (sin contar el ultimo): prime_range(n1,n2)
- Primo enésimo: nth_prime(n)
- Iterador de primos (para bucles): primes(n1,n2)
- Devolver un número racional con d digitos de precisión: numero.n(digits=d)