Skolan för Datavetenskap och kommunikation

PROGRAMMERINGSTEKNIK

FÖRELÄSNING 11

REKURSION, ALGORITMER, SIMULERING

REKURSIV TANKE

Ett problem där vi kan hitta lösningen genom att lösa en mindre variant av samma problem kan lösas rekursivt.

Exempel - skriv ut en sträng lodrätt.

Rekursiv tanke:

- Skriv ut första tecknet `s[0]`
- Skriv sen ut resten av strängen `s[1:]`

REKURSIV FUNKTION - TESTA

```
def rprint(s):
    print(s[0])
    rprint(s[1:])


rprint("sirap")
```

BASFALL

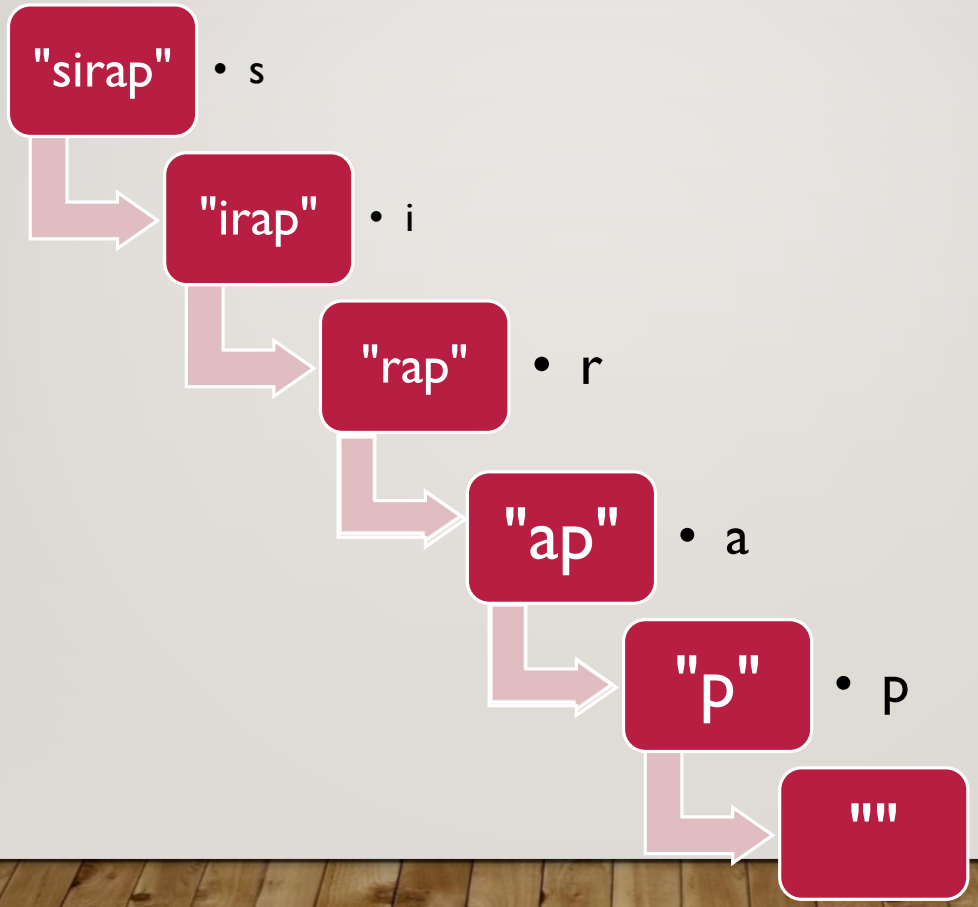
- Basfallet är det/de fall för vilket vi inte gör rekursivt anrop.
- Utan basfall får vi oändlig rekursion.
- Basfall för vår utskriftsfunktion?
  - En tom sträng ska inte skrivas ut.
  - Alltså kollar vi om strängen är tom i början av funktionen!

MED BASFALL

```
def rprint(s):
    if len(s) > 0:
        print(s[0])
        rprint(s[1:])

rprint("sirap")
```

VAD ÄR PARAMETERN S?

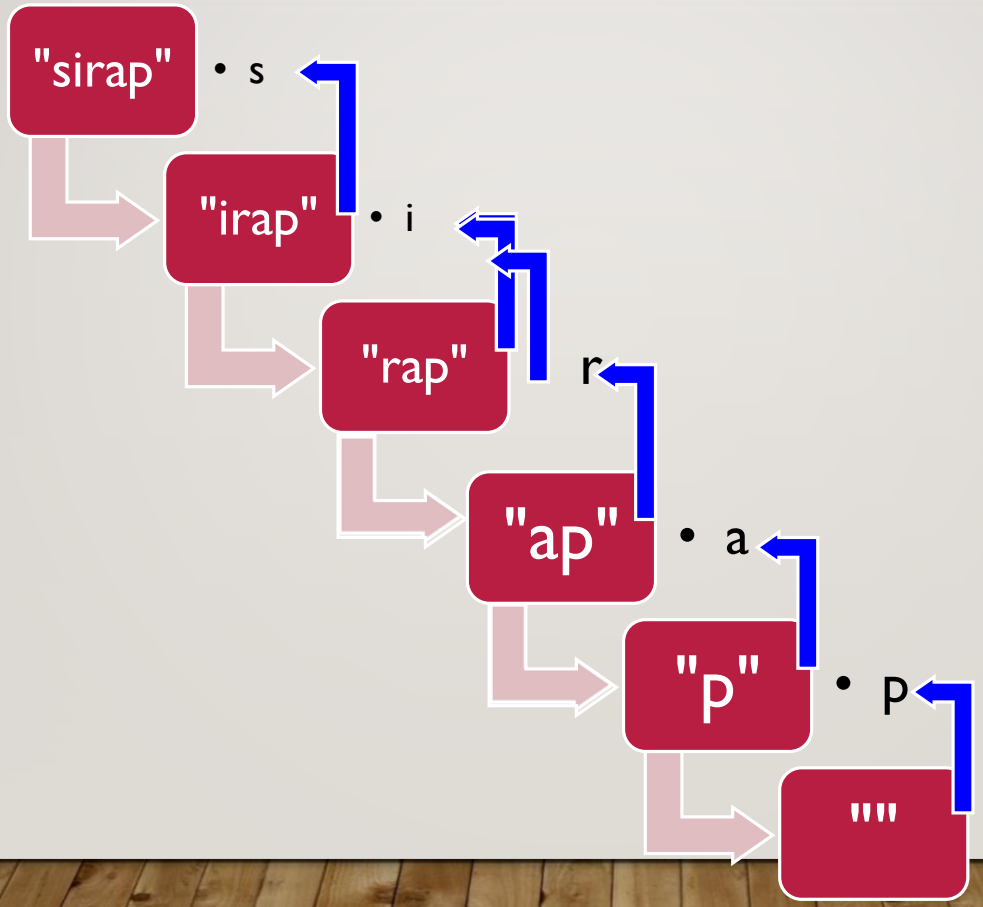


VAD HÄNDER HÄR?

```
def rprint(s):
    if len(s) > 0:
        rprint(s[1:])
        print(s[0])

rprint("sirap")
```

UTSKRIFT PÅ VÄG TILLBAKA



VAD SKRIVS UT NU?

```
def rprint(s):
    if len(s) > 0:
        print(s[0])
        rprint(s[1:])
        print(s[0])


rprint("sirap")
```

TORNEN I HANOI

PROBLEM

- Flytta alla brickor från vänstra till högra pinnen.
- Större brickor får inte läggas på mindre.
- För att flytta n brickor krävs $2^n - 1$ steg...
- Vi söker en algoritm som löser problemet.

PROVA MED SMÅ N

- n=1:
  - Flytta brickan från vänstra till högra pinnen.
- n=2:
  - Flytta lilla brickan från vänstra till mellanpinnen.
  - Flytta stora brickan från vänstra till högra pinnen
  - Flytta lilla brickan från mellanpinnen till högra pinnen.
- n=3:
  - Kan vi utnyttja n=2?

N=3

- n=3:
  - Flytta 2 brickor till mellanpinnen (rekursivt)
  - Flytta stora brickan från vänstra till högra pinnen.
  - Flytta 2 brickor från mellanpinnen till högra pinnen.

GENERELL ALGORITM

- Flytta n-1 brickor till mellanpinnen (rekursivt)
- Flytta stora brickan från vänstra till högra pinnen.
- Flytta n-1 brickor från mellanpinnen till högra pinnen.

REKURSIV FUNKTION

```
def hanoi(n, vänster, mellan, höger):
    if n > 0:
        hanoi(n - 1, vänster, höger, mellan)
        höger.append(vänster.pop())
        hanoi(n - 1, mellan, vänster, höger)
```

ÄR REKURSION INEFFEKTIVT?

- JA: Att byta ut en slinga mot rekursion kan göra att programmet tar längre tid att köra.
- NEJ: Om den rekursiva varianten är enklare att formulera och programmera.
- JA: för Fibonaccitalen...












HUR MÅNGA REKURSIVA ANROP?

```
def fib(n):
    if n == 0 or n == 1:
        return 1
    else:
        return fib(n-1) + fib(n-2)
```

```
def fib(n):
    if n == 0 or n == 1:
        return 1
    f0 = 1
    f1 = 1
    for i in range(1,n):
        fn = f1 + f0
        f0 = f1
        f1 = fn
    return fn
```

SIMULERING

- Skapa en modell
- Skriv ett program med slump som simulerar modellen

TÄRNINGSKAST

```
def kasta():
    return random.randint(1,6)
```

REPRODUCERBARA RESULTAT?

- Använd random.seed för att få samma slumpföljd ur random.randint()
- random.seed(1) ger 2 5 1 3 1...
- random.seed(2) ger 1 1 1 3 2...

osv