\t

CSCI-UA 480.4: APS
Algorithmic Problem Solving

Instructor: Joanna Klukowska

created based on materials for this class by
Bowen Yu and materials shared by the authors of
the textbook Steven and Felix Halim

Divide and Conquer

Divide and Conquer

- solving problems by *dividing* them into smaller/simpler problems and then *conquering* the subproblems

- general approach:

    - divide the given problem into sub-problems (often using halves)
    - find solutions to the sub-problems (often by following the same divide and conquer approach)
    - combine the solutions of sub-problems into a solution for the given problem

- examples

    - quicksort, mergesort
    - binary search

- decrease and conquer - a version of divide and conquer algorithms in which only one sub-problem needs to be solved (as in binary search)

Challenge: Paying Off a Loan

Task

You use a bank loan to buy a car. You need to pay off the loan by paying d dollars for m months. The original value of the car is v and the bank charges the interest rate of i for any unpaid amount at the end of each month.

Given the values for v, i and m figure out what d needs to be. Determine d with two digits after the decimal point.

Challenge: Paying Off a Loan

Task

You use a bank loan to buy a car. You need to pay off the loan by paying d dollars for m months. The original value of the car is v and the bank charges the interest rate of i for any unpaid amount at the end of each month.

Given the values for v, i and m figure out what d needs to be. Determine d with two digits after the decimal point.

Solution using a bisection method

- we want d in a particular given range [a,b] = [0.0, $v \times (1 + i/100)$]
- we start with an estimate for d as $\frac{a+b}{2}$
- if the result is that we overpay, then try to decrease d, otherwise, increase d

a	b	d	estimate	action
0.010000	1000.000000	500.005000	underpaid	increase d
500.005000	1000.000000	750.002500	overpaid	decrease d
500.005000	750.002500	625.003750	overpaid	decrease d
500.005000	625.003750	562.504375	overpaid	decrease d
500.005000	562.504375	531.254688	overpaid	decrease d
500.005000	531.254688	515.629844	underpaid	increase d
515.629844	531.254688	523.442266	underpaid	increase d
523.442266	531.254688	527.348477	overpaid	decrease d
523.442266	527.348477	525.395371	overpaid	decrease d
523.442266	525.395371	524.418818	overpaid	decrease d
523.442266	524.418818	523.930542	overpaid	decrease d
523.442266	523.930542	523.686404	underpaid	increase d
523.686404	523.930542	523.808473	underpaid	increase d
523.808473	523.930542	523.869507	overpaid	decrease d
523.808473	523.869507	523.838990	overpaid	decrease d
523.808473	523.838990	523.823732	overpaid	decrease d
523.808473	523.823732	523.816102	overpaid	decrease d
523.808473	523.816102	523.812288	overpaid	decrease d
523.808473	523.812288	523.810380	overpaid	decrease d
523.808473	523.810380	523.809427	done	

```
int m, x;  double value, i;

scanf ("%lf %lf %d", &value, &i, &m);

double a = 0.01, b = (1+1/100)*value;
double d = (a+b)/2;
double paid , owed ;

do {
    paid = 0.0, owed = value;
    for (int month = 1; month <= m; month++) {
        paid += d;
        owed -= d;
        owed *= (1+i/100);

    }
    printf("%f\t%f\t%f", a, b, d);

    if (fabs(0.0 - owed) < 0.001){
        printf("\tdone\n");
    }
    else if ( owed < 0) {
        b = d;
        printf ("\toverpaid\tdecrease d\n");
    }
    else {// ( owed > 0)
        a = d;
        printf ("\tunderpaid\tincrease d\n");
    }
    d = (a+b)/2;

}
while (fabs(0.0 - owed) >= 0.001 ) ;
```