Work

Definition.
The work of a program (on a given input) is the sum total of all the operations executed by the program.

Packing and Encoding

The idea of packing is to store more than one data value in a machine word. The related idea of encoding is to convert data values into a representation requiring fewer bits.

Example: Encoding dates

- The string "`September 11, 2018`" can be stored in 18 bytes — more than two double (64-bit) words — which must moved whenever a date is manipulated.
- Assuming that we only store years between 4096 B.C.E. and 4096 C.E., there are about $365.25 \times 8192 \approx 3\,\text{M}$ dates, which can be encoded in $\lceil \lg(3 \times 10^6) \rceil = 22$ bits, easily fitting in a single (32-bit) word.
- But determining the month of a date takes more work than with the string representation.

Precomputation

The idea of precomputation is to perform calculations in advance so as to avoid doing them at "mission-critical" times.

Example: Binomial coefficients

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Computing the "choose" function by implementing this formula can be expensive (lots of multiplications), and watch out for integer overflow for even modest values of n and k.

Idea: Precompute the table of coefficients when initializing, and perform table look-up at runtime.

Inlining

The idea of inlining is to avoid the overhead of a function call by replacing a call to the function with the body of the function itself.

```
double square(double x) {
  return x*x;
}

double sum_of_squares(double *A, int n) {
  double sum = 0.0;
  for (int i = 0; i < n; ++i) {
    sum += square(A[i]);
  }
  return sum;
}
```

```
double sum_of_squares(double *A, int n) {
  double sum = 0.0;
  for (int i = 0; i < n; ++i) {
    double temp = A[i];
    sum += temp*temp;
  }
  return sum;
}
```