Lab2 Lib

Pedro Vidal

1. DP

1.1. Prob da Moeda

```
vector<int> dp(m + 1, INF), coins(n), ultima(m + 1, 0); dp[0] = 0;
// read and sort coins
for (int i = 0; i < n; i++){
  for (int j = 0; j <= m && j + coins[i] <= m; j++){ // moedas ilimitadas
  // for (int j = m; j >= 0; j--){ //moedas limitadas
    if (dp[j] + 1 < dp[j + moeda[i]]) ultima[j + moeda[i]] = moeda[i];
    dp[j + coins[i]] = min(dp[j + coins[i]], dp[j] + 1);
  }
}
dp[m] == INF ? cout << "Impossivel" :  cout << dp[m]; cout << endl;
// dp[m] == quant de moedas usadas para formar m
```

1.2. Prob da Mochila

```
vector<int> p(n), v(n), dp(cap+1, _INF); dp[0] = 0;
// read weight (p) and value (v)
int res = 0;
for (int i = 0; i < n; i++){
  for (int j = cap; j >= 0; j--){
    if (dp[j] == _INF || j + p[i] > cap) continue;
    dp[j + p[i]] = max(dp[j + p[i]], dp[j] + v[i]);
    res = max(res, dp[j + p[i]]);
  }
}
cout << res << endl;
```

1.3. LCS

```
vector<vector<int> > lcs(ta, vector<int>(tb, 0));
for (int i = 1; i < ta; i++){
  for (int j = 1; j < tb; j++){
    lcs[i][j] = (a[i - 1] == b[j - 1] ? lcs[i - 1][j - 1] + 1 :
    max(lcs[i-1][j], lcs[i][j-1]));
  }
}
cout << lcs[ta-1][tb-1] << endl;
```

1.4. Edit Distance

```
vector<vector<int> > edist(ta, vector<int>(tb, 0));
for (int i = 0; i < ta; i++){
  for (int j = 0; j < tb; j++){
    if (i == 0 || j == 0){ edist[i][j] = max(i, j); continue;}
    edist[i][j] = (a[i-1] == b[j-1] ? edist[i-1][j-1] : min({edist[i-1][j-1],
    edist[i-1][j], edist[i][j-1]}) + 1);
  }
}
cout << edist[ta-1][tb-1] << endl;
```

1.5. Longest Increasing Subsequence

```
vector<int> lis(vet.size() + 1, INF);
lis[0] = _INF;
int res = 0;
for (int i = 0; i < vet.size(); i++){
  int pos = lower_bound(lis.begin(), lis.end(), vet[i]) - lis.begin();
  lis[pos] = vet[i];
  res = max(res, pos);
}
```