Lab 7: Dynamic Programming

Thực hành

Exercise 1:

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

int max(int a, int b){ return a > b ? a : b; }

int dp[101];
int cut[101];

int solve(int price[], int n){
    if(n == 0) return 0;
    if(dp[n] >= 0) return dp[n];

    int q = -1;
    for(int i=1;i<=n;i++){
        int val = price[i-1] + solve(price, n-i);
        if(val > q){
            q = val;
            cut[n] = i;
        }
    }
    return dp[n] = q;
}

void print_cut(int n){
    printf("cut: ");
    while(n > 0){
        printf("%d ", cut[n]);
        n -= cut[n];
    }
    printf("\n");
}

int main(){
    int price[] = {1,5,8,9,10,17,17,20};
    int n = sizeof(price)/sizeof(price[0]);
    memset(dp, -1, sizeof(dp));
    int ans = solve(price, n);
    printf("max: %d\n", ans);
    print_cut(n);
    return 0;
}

#include<stdio.h>
int max(int a, int b){ return a > b ? a : b; }

void rod_bottom(int price[], int n){
    int dp[101] = {0};
    int cut[101] = {0};

    for(int j=1;j<=n;j++){
        int q = -1;
        for(int i=1;i<=j;i++){
            if(q < price[i-1] + dp[j-i]){
                q = price[i-1] + dp[j-i];
                cut[j] = i;
            }
        }
        dp[j] = q;
    }

    printf("max: %d\n", dp[n]);
    printf("cut: ");
    while(n > 0){
        printf("%d ", cut[n]);
        n -= cut[n];
    }
    printf("\n");
}

int main(){
    int price[] = {1,5,8,9,10,17,17,20};
    int n = sizeof(price)/sizeof(price[0]);
    rod_bottom(price, n);
    return 0;
}
```

Exercise 2:

```
#include<stdio.h>

int max(int a, int b){ return a > b ? a : b; }

void rod_bottom(int price[], int n){
    int dp[101] = {0};
    int cut[101] = {0};

    for(int j=1;j<=n;j++){
        int q = -1;
        for(int i=1;i<=j;i++){
            if(q < price[i-1] + dp[j-i]){
                q = price[i-1] + dp[j-i];
                cut[j] = i;
            }
        }
        dp[j] = q;
    }

    printf("max: %d\n", dp[n]);
    printf("cut: ");
    while(n > 0){
        printf("%d ", cut[n]);
        n -= cut[n];
    }
    printf("\n");
}

int main(){
    int price[] = {1,5,8,9,10,17,17,20};
    int n = sizeof(price)/sizeof(price[0]);
    rod_bottom(price, n);
    return 0;
}
```

Exercise 3:

```
#include <stdio.h>
#include <string.h>

int memo[101][101]; // hope 100 is enough
char str1[101], str2[101];

int lcs(int i, int j) {
    if (i == 0 || j == 0)
        return 0;
    if (memo[i][j] != -1)
        return memo[i][j];
    if (str1[i-1] == str2[j-1])
        memo[i][j] = 1 + lcs(i-1, j-1);
    else
        memo[i][j] = (lcs(i-1, j) > lcs(i, j-1)) ? lcs(i-1, j) : lcs(i, j-1);
    return memo[i][j];
}

void printLCS(int i, int j) {
    if (i == 0 || j == 0) return;
    if (str1[i-1] == str2[j-1]) {
        printLCS(i-1, j-1);
        printf("%c", str1[i-1]);
    } else if (memo[i-1][j] > memo[i][j-1]) {
        printLCS(i-1, j);
    } else {
        printLCS(i, j-1);
    }
}

int main() {
    scanf("%s", str1);
    scanf("%s", str2);
    int n = strlen(str1);
    int m = strlen(str2);

    for (int i = 0; i <= n; i++)
        for (int j = 0; j <= m; j++)
            memo[i][j] = -1;

    int len = lcs(n, m);
    printf("LCS length = %d\n", len);
    printf("LCS = ");
    printLCS(n, m);
    printf("\n");

    return 0;
}
#include <stdio.h>
#include <string.h>

int dp[101][101]; // maybe too big lol
char str1[101], str2[101];

void printLCS(int n, int m) {
    int i = n, j = m;
    char lcs[101];
    int idx = 0;
    while (i > 0 && j > 0) {
        if (str1[i-1] == str2[j-1]) {
            lcs[idx++] = str1[i-1];
            i--;
            j--;
        } else if (dp[i-1][j] >= dp[i][j-1]) {
            i--;
        } else {
            j--;
        }
    }
    // reverse the string
    printf("LCS = ");
    for (int k = idx - 1; k >= 0; k--)
        printf("%c", lcs[k]);
    printf("\n");
}
```

```
int main() {
    scanf("%s", str1);
    scanf("%s", str2);
    int n = strlen(str1);
    int m = strlen(str2);

    for (int i = 0; i <= n; i++) {
        for (int j = 0; j <= m; j++) {
            if (i == 0 || j == 0)
                dp[i][j] = 0;
            else if (str1[i-1] == str2[j-1])
                dp[i][j] = dp[i-1][j-1] + 1;
            else {
                if (dp[i-1][j] > dp[i][j-1])
                    dp[i][j] = dp[i-1][j];
                else
                    dp[i][j] = dp[i][j-1];
            }
        }
    }

    printf("LCS length = %d\n", dp[n][m]);
    printLCS(n, m);

    return 0;
}
```