B03902062 資工二 董文捷

For number of combinations, I use dynamic programming to calculate it. Each time add a new kind of coin, if m dollars has n ways to pay, we can add one coin[i] and pay m + coin[i] dollars.

My code in C++:

```
#include <stdio.h>
const int money_max = 30;
int main()
{
    int i, j;
    int coins[4] = {1, 5, 10, 50};
    int combination_num[money_max] = {0};
    combination_num[0] = 1;
    for(i = 0; i < 4; i++)
        for(j = 0; j < money_max - coins[i]; j++)
            combination_num[j + coins[i]] += combination_num[j];
    printf("%d\n", combination_num[22]);
    return 0;
}
```

ANS : 9

To pay 22 dollars in the least number of coins, we can use greedy because each smaller coin is a factor of bigger coin

22 / 50 = 0 22 / 10 = 2

2 / 5 = 0 2 / 1 = 2

ANS: at least 4 coins