

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
JNANA SANGAMA,BELAGAVI – 590018
KARNATAKA



Mini Project Report
On

**“Coin Change Problem using Brute Force
and Greedy Algorithm”**

SUBMITTED IN PARTIAL FULFILLMENT OF THE ASSIGNMENT
FOR THE **Analysis & Design of Algorithms (BCS401)**
COURSE OF IV SEMESTER

Submitted by

Name: Amith A N

USN: 1CG22CS008

Name: Jayasurya N

USN: 1CG22CS047

Guide:

Mr.Asif Ulla Khan,M. Tech.
Asst. Prof., Dept. of CSE
CIT, Gubbi.

HOD:

Dr. Shantala C PPhD.,
Head, Dept. of CSE
CIT, Gubbi.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



Channabasaveshwara Institute of Technology

(Affiliated to VTU, Belgaum & Approved by AICTE, New Delhi)

(NAAC Accredited & ISO 9001:2015 Certified Institution)

NH 206 (B.H. Road), Gubbi, Tumkur – 572 216. Karnataka



2023-24

Coin Change Problem using Brute Force and Greedy Algorithm – B.E.

Mini-Project [BCS401]

Course outcome	Rubric/Level	Excellent (91-100%)	Good (81-90%)	Average (61-80%)	Moderate (40-60%)	Score
CO1	Identification of project proposal (05 Marks)					05
CO2	Design and Implementation (05 Marks)					05
CO3	Presentation skill (05 Marks)					05
CO 4	Individual or in a team development					05
CO5	Report (05 Marks)					05
Total						25

Course outcome:

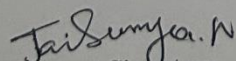
CO 1: Identification of project proposal which is relevant to subject of engineering.

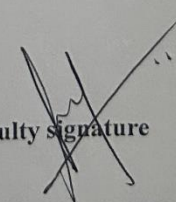
CO 2: Design and implement proposed project methodology.

CO 3: Effective communication skill to assimilate their project work.

CO 4: Work as an individual or in a team in development of technical projects.

CO 5: Understanding overall project progress and performance.


Student Signature


Faculty Signature

Coin Change Problem using Brute Force and Greedy Algorithm – B.E.
Mini-Project [BCS401]

Course outcome	Rubric/Level	Excellent (91-100%)	Good (81-90%)	Average (61-80%)	Moderate (40-60%)	Score
CO1	Identification of project proposal (05 Marks)					05
CO2	Design and Implementation (05 Marks)					05
CO3	Presentation skill (05 Marks)					05
CO 4	Individual or in a team development					05
CO5	Report (05 Marks)					05
Total						05

Course outcome:

CO 1: Identification of project proposal which is relevant to subject of engineering.

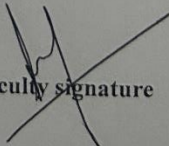
CO 2: Design and implement proposed project methodology.

CO 3: Effective communication skill to assimilate their project work.

CO 4: Work as an individual or in a team in development of technical projects.

CO 5: Understanding overall project progress and performance.


 Student Signature


 Faculty signature

ABSTRACT

Automatic money exchange systems are very rarely found but are needed at certain times. The purpose of this project is to make it easier for people who want to exchange money from big to small nominal. The obstacle faced in this exchange system is the use of algorithms needed to run the system. Some method that can be used to design and implement this system is using Greedy Algorithm and Brute force algorithms. By using this greedy algorithm, a problem with the shortest route search technique can be completed quickly, but the results generated by the greedy algorithm are not always optimal, while brute force can solve the problem of money exchange optimally but need a longer time. In this subject we trying to solve this problem by making an application using python language to implement the greedy and brute force algorithm in solving of this problem. After making the code algorithm and implementing the greedy and brute force method, we conclude that our program can solve this coins exchange problem just in a second but sometimes the result is not optimal (with the most smallest fraction), but if using brute force algorithm we can get the optimal total of money changes but the time will extremely increase up to 20 times higher depend on the amount of the total money that want to be changes in this experiment the user will input the money that want to change, in the program we already have a predetermined amount and money that has been prepared to exchange money.

CHAPTER 1:

INTRODUCTION

To solve a problem of course it has a way or solution so that the problem is resolved properly. Any problem of course has a solution. In a research solution is needed to help solve a problem. Simillar to this report which will discuss about greedy algorithm and brute force. In daily life we always deal with financial problems, one of the problem which is how the cashier gives change money. This report will discuss the comparison of Greedy and Brute Force algorithm in the example of optimization problem, namely the problem of money exchange. In this case we want to see and know which algorithm has an optimum result and more fast. In this experiment we will exchange money in rupiah currency. In this case we will Exchange money with a predetermined amount and money that has been prepared to exchange money. The money will be exchanged using smaller money and will give the minimum and equivalent to the money that has been exchanged. The method used to solve this coin-change problem is Brute Force and Greedy Algorithm which will be used for comparison.

CHAPTER 2:

PROBLEM STATEMENT

You are a cashier at a store in Indonesia, and you need to make change for a customer in Rupiah coins. You have coins of denominations 100, 200, 500, and 1000 Rupiah. Write a program to find the minimum number of coins needed to make the change using both a brute force approach and a greedy algorithm.

Input:

The amount of Rupiah to make change for (an integer)

Output:

The minimum number of coins needed to make the change using the brute force approach

The minimum number of coins needed to make the change using the greedy algorithm

The time taken to execute each approach

Constraints:

The amount of Rupiah to make change for is a positive integer

The coin denominations are 100, 200, 500, and 1000 Rupiah

The program should be able to handle amounts up to 10,000 Rupiah

Example Input/Output:

Input: 1200 Rupiah

Output:

Brute Force: 3 coins (1000 + 200)

Greedy Algorithm: 3 coins (1000 + 200)

Time taken: 0.01 seconds (brute force), 0.005 seconds (greedy algorithm)

CHAPTER 3:

IMPLEMENTATION

Analysis of Algorithm:

Greedy Algorithm :The Greedy algorithm is the algorithm most often used in solving optimization problems. Greedy itself is taken from English, which means “serakah”, “tamak” or “rakus”. In accordance with these meanings, the Greedy principle is "Take what you can get now". The greedy algorithm will form a step-by-step solution. At each step, we must make choices that can produce optimum results. Greedy is an algorithmic paradigm that builds up a solution piece by piece, always choosing the next piece that offers the most obvious and immediate benefit. So the problems where choosing locally optimal also leads to global solution are best fit for Greedy.

Brute Force Algorithm :Brute Force Algorithms refers to a programming style that does not include any shortcuts to improve performance, but instead relies on sheer computing power to try all possibilities until the solution to a problem is found. The Brute force algorithm is a straightforward approach to solving a problem, usually based on a problem statement. The brute force algorithm solves problems very simply, directly and in a clear way (obvious way).

Algorithm:

Coin Change Problem works based on the “**Brute force and Greedy**” algorithm.

Brute Force Algorithm:

Step 1: Define the problem

Input: The amount of Rupiah to make change for (an integer)

Output: The minimum number of coins needed to make the change

Coin denominations: 100, 200, 500, and 1000 Rupiah

Step 2: Generate all possible combinations of coins

Start with an empty combination: []

Iterate over each coin denomination in descending order (1000, 500, 200, 100)

For each coin denomination, generate all possible combinations of coins that sum up to the input amount

For example, if the input amount is 1500, the possible combinations for the 1000 Rupiah coin are:

$$1 \times 1000 = 1000$$

$$2 \times 1000 = 2000 \text{ (exceeds the input amount, so skip)}$$

For each possible combination, recursively generate all possible combinations of coins for the remaining amount

For example, if the remaining amount is 500, the possible combinations for the 500 Rupiah coin are:

$$1 \times 500 = 500$$

$$2 \times 500 = 1000 \text{ (exceeds the remaining amount, so skip)}$$

Continue generating combinations until the remaining amount is 0

Step 3: Evaluate each combination

For each generated combination, calculate the total number of coins used

Keep track of the combination with the minimum number of coins used

Step 4: Return the minimum number of coins needed

Return the minimum number of coins needed to make the change, which is the length of the combination with the minimum number of coins used

Greedy Algorithm:

Step 1: Define the problem

Input: The amount of Rupiah to make change for (an integer)

Output: The minimum number of coins needed to make the change

Coin denominations: 100, 200, 500, and 1000 Rupiah

Step 2: Sort the coin denominations in descending order

Sort the coin denominations in descending order: 1000, 500, 200, 100

Step 3: Iterate over each coin denomination

For each coin denomination, calculate how many times it can be used to make the change

For example, if the input amount is 1500, the 1000 Rupiah coin can be used 1 time ($1000 \times 1 = 1000$)

Update the remaining amount: $1500 - 1000 = 500$

Repeat step 3 until the remaining amount is 0

Step 4: Return the minimum number of coins needed

Return the total number of coins used, which is the sum of the number of times each coin denomination was used

PROGRAM:

```
import sys
import time
def menu():
    print("          Main Menu\n")
    print("  1. Rupiah Coin Change by Brute Force\n")
    print("  2. Rupiah Coin Change by Greedy Algorithm\n")
    print("  0. Exit\n")
def getNumOfCoins(coins, sum):
    if (sum == 0):
        return 0
    else:
        result = sys.maxsize # max integer
        for coin in coins:
            if (coin <= sum):
                result = min(result, getNumOfCoins(coins, sum - coin) + 1)
        return result
def greedy(exchange):
    jmlh7 = 0
    jmlh5 = 0
    jmlh3 = 0
    jmlh1 = 0
    while(exchange != 0):
        if(exchange >= 1000):
            exchange = exchange - 1000
            jmlh7 = jmlh7 + 1
        elif(exchange >= 500):
            exchange = exchange - 500
            jmlh5 = jmlh5 + 1
        elif(exchange >= 200):
            exchange = exchange - 200
            jmlh3 = jmlh3 + 1
        else:
            exchange = exchange - 100
            jmlh1 = jmlh1 + 1
```

```

print("    Jumlah coin 1000 :", jmlh7)
print("    Jumlah coin 500 :", jmlh5)
print("    Jumlah coin 200 :", jmlh3)
print("    Jumlah coin 100 :", jmlh1)
if __name__ == "__main__":
    coins = [100, 200, 500, 1000]
    stsum = input("Enter Rupiah amount to change in coin: ")
    sum = int(stsum)
    menu()
    val = input("    Enter your choice: ")
    while val != '0':
        if val == '1':
            print("    Coins : ", coins)
            print("    Sum : ", sum)
            begin = time.perf_counter()
            print("    Minimum coins: ", getNumOfCoins(coins, sum))
            end = time.perf_counter()
            print("    Time execution : ", end - begin)
        elif val == '2':
            begin = time.perf_counter()
            greedy(sum)
            end = time.perf_counter()
            print("    Time execution : ", end - begin)
        menu()
        val = input("    Enter your choice: ")

```

Pseudo-code :

a) Brute Force

Function getNumOfCoins(coins, sum)

If (sum=0) then

Return 0

Else

result = max integer

coin=0

For (coin<coins.length) do //array start at 0

```
if (coins[coin]<=sum) then
    result = min(result,getNumOfCoins(coins,sum-coin)+1)
```

b) Greedy Algorithm

Function greedy(exchange)

Jmlh7 = 0

Jmlh5 = 0

Jmlh3 = 0

Jmlh1 = 0

While(exchange \neq 0) do

If(exchange \geq 1000) then

Exchange = exchange-1000

Jmlh7=jmlh7+1

Else if(exchange \geq 500) then

Exchange = exchange-500

Jmlh5=jmlh5+1

Else if(exchange \geq 200) then

Exchange = exchange-200

Jmlh3=jmlh3+1

Else

Exchange= exchange-100

Jmlh1=jmlh1+1

Endwhile

Print("Jumlah coin 1000:", jmlh7)

Print("Jumlah coin 500:", jmlh5)

Print("Jumlah coin 200:", jmlh3)

Print("Jumlah coin 100:", jmlh1)

Analyse the time:

Brute force:

Number of Coins in the array: 4

The number of coins to be exchanged: n

Time Complexity: 4^n

Greedy Algorithm:

Number of Coins in the array: 4

The number of coins to be exchanged: n

Time complexity: N

CHAPTER 4:

RESULTS

Brute force Algorithm:

```
Enter Rupiah amount to change in coin: 3000
Main Menu

1. Rupiah Coin Change by Brute Force

2. Rupiah Coin Change by Greedy Algorithm

0. Exit

Enter your choice: 1
Coins : [100, 200, 500, 1000]
Sum : 3000
Minimum coins: 3
Time execution : 3.521698652068153
```

Greedy Algorithm:

```
Enter Rupiah amount to change in coin: 3000
Main Menu

1. Rupiah Coin Change by Brute Force

2. Rupiah Coin Change by Greedy Algorithm

0. Exit

Enter your choice: 2
Jumlah coin 1000 : 3
Jumlah coin 500 : 0
Jumlah coin 200 : 0
Jumlah coin 100 : 0
Time execution : 5.670600512530655e-05
```

```
Enter Rupiah amount to change in coin: 3000
Main Menu

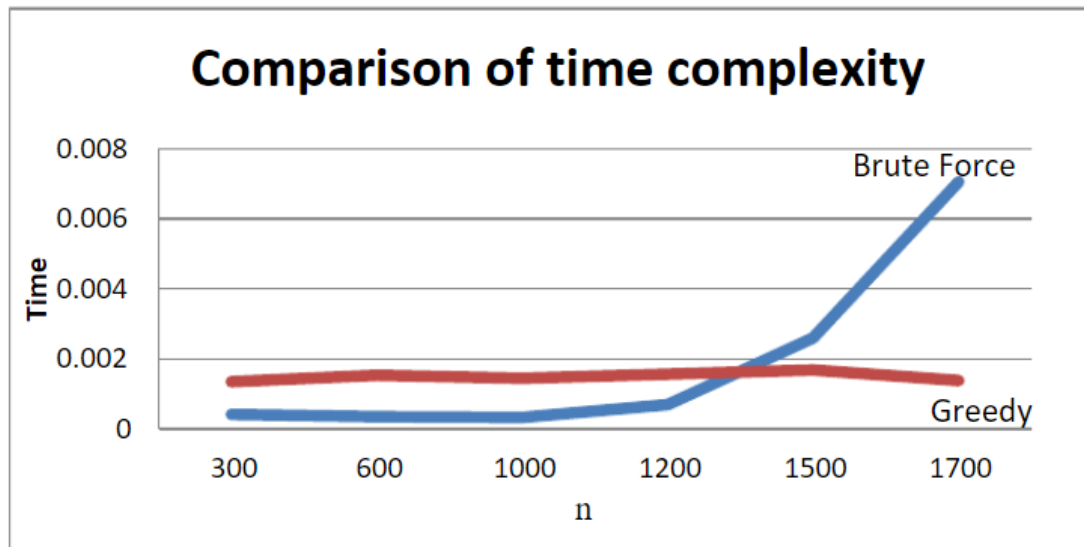
1. Rupiah Coin Change by Brute Force

2. Rupiah Coin Change by Greedy Algorithm

0. Exit

Enter your choice: 0

...Program finished with exit code 0
Press ENTER to exit console.
```



From graph 1, the greedy algorithm (red) and brute force (blue) have different time executions. as in the example of exchanging money for Rp.1,700, for bruteforce time in problem solving is greedy algorithm almost 7x faster to solve the coin change problem when greedy is done. From this experiment it can be concluded that the greedy algorithm is faster even though the results are not always optimum. For Brute force the more money will be exchanged, the longer the execution time will be, while the Greedy Algorithm has an execution time that is faster than brute force.

CHAPTER 5:

CONCLUSION

From the comparison of the Greedy algorithm and brute force in our experiments, it can be concluded that the greedy algorithm can produce results that are faster than brute force, in example on the total coin change 1700 rupiah, greedy algorithm almost 7x faster to solve the coin change problem when greedy done to execute the program in 0.0013787 second and bruteforce done in 0.0070634 second. However, the results of the money exchange from the greedy algorithm cannot always produce optimal results.

REFERENCES:

- <https://www.blackbox.ai/>
- <https://dosenit.com/ilmu-komputer/pengertian-algoritma-brute-force-dan-greedy>