1. (16 points) Let $P_1, P_2, \ldots, P_n$ be a set of $n$ programs that are to be stored on a tape of length $l$. Program $P_i$ requires $a_i$ amount of tape. If $\sum a_i \leq l$, then clearly all the programs can be stored on the tape. So, assume $\sum a_i > l$. The problem is to select a maximum subset $Q$ of the programs for storage on the tape. (A maximum subset is one with the maximum number of programs in it). Present a greedy algorithm for this problem and prove that this algorithm will always find an optimal solution. What is the run time of your algorithm? Let $Q$ be the subset obtained using your algorithm. How small can the tape utilization ratio $(\sum_{P_i \in Q} a_i)/l$ get?
2. Consider the integer knapsack problem obtained by replacing the 0/1 constraint by \( x_i \geq 0 \) and integer. Generalize \( f_i(y) \) to this problem in the obvious way. (Note that \( f_i(y) \) is the maximum profit to the problem \( \text{KNAP}(1, i, y) \).)

(a) (8 points) Obtain the dynamic programming recurrence relation corresponding to \( f_i(y) \).

(b) (4 points) Show how to solve this relation in \( O(m^2n) \) time assuming that the weights are integers.

(c) (4 points) Show that the real knapsack problem with \( x_i \geq 0 \) can be solved in \( O(n) \) time.
3. (18 points) Let $A_n = \{a_1, a_2, \ldots, a_n\}$ be a finite set of distinct coin types (for example, $a_1 = 50\$, $a_2 = 25\$, $a_3 = 10\$, and so on.) We can assume each $a_i$ is an integer and $a_1 > a_2 > \cdots > a_n$. Each type is available in unlimited quantity. The coin-changing problem is to take an integer $C$ as input and make up an exact amount $C$ using a minimum total number of coins. Assume that $a_n = 1$ so that there is always a solution. Present an $O(Cn)$ time algorithm for this problem. \textit{Hint: Use dynamic programming.}
4. (18 points) Design an algorithm to decide whether a given undirected graph \( G(V,E) \) contains a square (i.e., a cycle of length 4) as a subgraph. The running time of the algorithm should be \( O(|V||E|) \). You can use the adjacency list representation or the adjacency matrix representation, whichever is more convenient.
5. (15 points) Let \( f(x) \) and \( g(x) \) be two polynomials of degree \( n \) each. Also let \( h(x) = f(x) \times g(x) \).

Present an \( O(n) \) time algorithm that will take as input \( f \) and \( g \) (in coefficients form) and compute \( h(a) \) where \( a \) is a scalar.
6. (17 points) Present an algorithm to compute $A$ and $B$ are sets of integers in the range $[0,5n]$. Define $C_i = \{(x,y) : x \in A, y \in B, & x + y = i\}$, for $i = 0, 1, \ldots, 10n$. Present an $O(n \log n)$ time algorithm to compute $|C_i|$, for $i = 0, 1, \ldots, 10n$. 