7) Design and implement C/C++ Program to solve discrete Knapsack and continuous Knapsack problems using greedy approximation method.

Soln:

```
#include <stdio.h>

#include <conio.h>


void Gknapsack();

int max(int, int);

int i, j, q, n, m1, p[10], w[10], sol[10];

float ratio[10];


void main() {

  printf("\n Enter the no.of items:\n");

  scanf("%d", &n);


  printf("\n Enter the weight of each item:\n");

  for (i = 0; i < n; i++) {

    scanf("%d", &w[i]);

  }


  printf("\n Enter the profit of each item:\n");

  for (i = 0; i < n; i++) {

    scanf("%d", &p[i]);

  }
  printf("\n Enter the knapsack's capacity:\t ");
  scanf("%d", &m1);

  Gknapsack();
}

void Gknapsack() {
  int sum = 0;

  for (i = 0; i < n; i++) {
    ratio[i] = (float)p[i] / w[i];
    sol[i] = 0; // Initialize solution array to 0
  }

  for (q = 0; q < n; q++) {
    float max = 0.0;
    int k = −1;

    for (i = 0; i < n; i++) {
      if ((ratio[i] > max) && (sol[i] == 0)) {
        max = ratio[i];
        k = i;
      }
    }

    if (k == −1) break; // If no item found, break the loop
```

```
            if (m1 >= w[k]) {

                sol[k] = 1;

                m1 -= w[k];

                sum += p[k];

            } else {

                sol[k] = -1;

            }

        }


        for (i = 0; i < n; i++) {

            if (sol[i] == -1) {

                sol[i] = 0;

            }

        }


        printf("\n Solution with Greedy Technique: %d", sum);

        printf("\nThe solution vector is :\n");

        printf("[");


        for (i = 0; i < n; i++) {

            printf("%d\t", sol[i]);

        }


        printf("]");

    }
```