cs330su21-a.sg

Grade

Reviewed on Friday, 30 July 2021, 2:35 AM by Automatic grade
grade: 100.00 / 100.00

Assessment report [-]
[+]**Summary of tests**

Submitted on Friday, 30 July 2021, 2:35 AM (Download)
q.cpp

```cpp
/*!*************************************************************************
\file    q.cpp
\author  Goh Wei Zhe
\par     DP email: weizhe.goh@digipen.edu
\par     Course: CS330
\par     Section: A
\par     Programming Assignment #3
\date    29-07-2021

\brief   Dynamic Programming, 0-1 Knapsack, Coin Changes, Tree Tower
*************************************************************************/

// only include the following two head files
#include <iostream>
#include <vector>
#include <algorithm> // max number from a vector
// Don't add more

#define INT_MAXI (int) ((unsigned) -1 / 2)
#define INT_MINI -INT_MAXI - 1

namespace CS330
{
  namespace dp
  {
      /*!*********************************************************************
      \brief
      Given that weight limit as a constraint and a list of the charms with
      their weights and desirability rating, deduce the maximum possible sum of
      ratings.

      \param W
      An array vector of weights

      \param D
      An array vector of desirable ratings

      \return
      Returns the maximum possible sum of desirability ratings.
      *********************************************************************/
      int charm_bracelet(int M, std::vector<int> const& W,
          std::vector<int> const& D)
      {
          int size = static_cast<int>(W.size());
          std::vector<int> Bag(M + 1);

          for (int i = 0; i < size; ++i)
          {
              for (int j = M; j >= W[i] ; --j)
              {
                  Bag[j] = std::max(Bag[j], Bag[j - W[i]] + D[i]);
              }
          }

          return Bag[M];
      }

      /*!*********************************************************************
      \brief
      Find the minimum numbers of coins for a specific change value.

      \param value
      Value to find minimum of coin change for

      \param denominations
      A vector of coins denominations

      \return
      Returns the minimum number of coins required to form the change value.
      *********************************************************************/
      int coin_changes(int value, std::vector<int> const& denominations)
      {
          int size = value + 1;

          //Set all values to max value
          std::vector<int> table(size, INT_MAXI);

          //Init first element to 0
          table[0] = 0;

          //Iterate through all values
          for (int i = 1; i <= value; ++i)
          {
              //Iterate through all demonimations
              for (const auto& x : denominations)
              {
                  int index = i - x;
                  if (i >= x && (table[index] != INT_MAXI))
                  {
                      table[i] = std::min(table[i], table[index] + 1);
                  }
              }
          }

          if (table[value] == INT_MAXI)
              return -1;

          return table[value];
      }

      /*!*********************************************************************
      \brief
      Find a path from root node to one of the leaf nods that has maximal total
      weight.

      \param rows
      Number of rows

```

```
    \param cols
    Number of cols

    \param weights
    A pointer to an int array

    \return
    Returns the maximal total weight from root node to leaf node among all
    paths.
    ***********************************************************************/
    int tree_tower(int rows, int cols, int* weights)
    {
        int size = rows * cols;

        //Iterate from -1 row and -1 colum, bottom up approach
        for (int i = size - rows - 1; i >= 0; i--)
        {
            //Get maximum value between next row index value and
            //next row + 1 index value
            weights[i] = std::max(weights[i + cols],
                                  weights[i + cols + 1]) + weights[i];
        }

        return weights[0];
    }
  }
}
```

VPL

◄ Attendance cs330su21-a.sg Thursday 22/07/2021 11:00am-12:40pm

Jump to...

Attendance cs330su21-a.sg Tuesday 27/07/2021 11:00am-12:40pm ►