САА - Lab 5

Problem

The following function finds the greatest single number within an array, but has an efficiency of O(N2). Rewrite the function so that it becomes a speedy O(N):

int greatestNumber(int array[], int size){

bool isIValTheGreatest;

for(int i = 0; i<size; i++){

isIValTheGreatest = true;

for (int j = 0; j<size; j++){

if(array[j] > array[i]){

isIValTheGreatest = false;

}

}

if(isIValTheGreatest==true){

return array[i];

}

}

}

Problem

The following function returns whether or not a capital “X” is present within a string.

bool containsX(string str){

bool foundX = false;

for(int i = 0; i < str.length(); i++){

if (str[i] == 'X'){

foundX = true;

}

}

return foundX;

}

This algorithm has an efficiency of O(N), modify the code to improve the algorithm’s efficiency for best- and average-case scenarios.

Problem

Create a new function to reverse an array that takes up just O(1) extra space.

VIII.Greedy Algorithms

Problem

Using a greedy algorithm to find what is the smallest number of coins you need to use to pay a certain amount of money (iterative version).

#include <iostream>

using namespace std;

const int num = 6;

int coins[num] = {50,20,10,5,2,1};

void count\_coins(int sum)

{

int i, j;

j = sum;

for(i=0; i<num; i++)

{

cout << "Coin: " << coins[i] << ", number = " << j/coins[i]<<endl;

j = j % coins[i];

}

}

int main()

{

int sum;

cout << "Enter a sum: ";

cin >> sum;

count\_coins(sum);

return 0;

}

Problem

Solve the previous problem using recursion.

IX.Divide-and-Conquer

Problem

Tower of Hanoi.

#include <iostream>

using namespace std;

const unsigned N = 3;

void diskMove(unsigned N, char a, char b)

{

cout << N << a << b << endl;

}

void hanoi(char source, char dest, char aux, unsigned numb)

{

if (1 == numb)

diskMove(1, source, dest);

else

{

hanoi(source, aux, dest, numb-1);

diskMove(numb, source, dest);

hanoi(aux, dest, source, numb-1);

}

}

int main()

{

cout << N << endl;

hanoi('A', 'C', 'B', N);

return 0;

}