Tema

NICOLETA RADU

1. Se dă a, b și c, să se calculeze a^b^c  modulo 10^9+7.

Solutie:

int main()

{

int a{ 0 }, b{ 0 }, c{ 0 }, result{0}, secondResult{0};

cout << "Enter number for variable a: ";

cin >> a;

cout << "Enter number for variable b: ";

cin >> b;

cout << "Enter number for variable c: ";

cin >> c;

result = pow(a, b);

cout << "a^b is equal to: " << result << endl;

result = pow(result, c);

cout << "a^b^c is equal to: " << result << endl;

secondResult = pow(10, 9);

cout << "10^9 + 7 is equal to: " << secondResult << endl;

cout << "a^b^c % 10^9 + 7 = " << (result % secondResult) + 7 << endl;

return 0;

}

1. Se dă un șir cu n elemente, **numere naturale** și un număr k. Să se determine câte secvențe din șir au **lungimea** k și sunt formate din valori **mai mici sau egale** cu t, unde t este **ultimul element** al șirului.

Solutie:

void printVector(vector <int> yourVector)

{

cout << endl;

cout << "Your vector: " << endl;

cout << "[";

for (size\_t i = 0; i < yourVector.size(); i++)

{

cout << yourVector.at(i) << ",";

}

cout << "]";

cout << endl;

}

void addValuesToVector(vector <int>& yourVector, int vectorSize)

{

for (size\_t i = 0; i < vectorSize; i++)

{

cout << "Value at index " << i << ": ";

cin >> yourVector.at(i);

}

}

void addValueToVar(int &yourVar)

{

cout << "Enter a value for the variable: ";

cin >> yourVar;

}

void printVar(int yourVariable)

{

cout << "Your variable has the value: "<< yourVariable;

}

int main()

{

int n{ 0 };

cout << "Vector size" << endl;

addValueToVar(n);

vector <int> myVector(n,0);

int k{ 0 }, t{ 0 }, count{ 0 }, aux{0};

addValuesToVector(myVector, n);

printVector(myVector);

cout << "desired sequence length" << endl;

addValueToVar(k);

t = myVector.at(n - 1);

cout << endl;

printVar(t);

for (size\_t i = 0; i < myVector.size() - 1; i++)

{

if (myVector.at(i) <= t && aux < k)

{

cout << "(in if) the value is: " << myVector.at(i) << " at index " << i << endl;

aux++;

cout << "(in if) aux is: " << aux << endl;

if (aux == k)

{

count++;

cout << "(in if) count is: " << count << endl;

cout << "(in if) aux is (should be equal to k): " << aux << endl;

aux = 0;

cout << "(in if) aux is (should be 0): " << aux << endl;

}

}

}

cout << "count is: " << count << endl;

return 0;

Solutie 2 (diferita) :

for (size\_t i = 0; i < myVector.size() - 1; i++)

{

for (size\_t j = i + 1; j < myVector.size(); j++)

{

cout << " THE VALUE IS: " << myVector.at(i) << " at index " << i << endl;

cout << endl;

if (myVector.at(i) <= t && myVector.at(j) <= t && aux < k - 1)

{

cout << "(in if) the value is: " << myVector.at(j) << " at index " << j << endl;

cout << endl;

aux++;

cout << "(in if) aux is: " << aux << endl;

if (aux == k - 1)

{

count++;

cout << "count is: " << count << endl;

j = n;

cout << "aux is (should be equal to k): " << aux << endl;

aux = 0;

cout << "aux is (should be 0): " << aux << endl;

cout << endl;

}

}

else

{

aux = 0;

j = n;

}

}

}

cout << "count is: " << count << endl;

return 0;



1. Se dau n numere naturale. Aflati daca acestea pot fi laturile unui poligon cu n laturi.

Solutie:

int findMaxVar(int firstVariable, int secondVariable)

{

if (firstVariable > secondVariable)

{

return firstVariable;

}

else

{

return secondVariable;

}

}

int main()

{

cout << "Vector size";

int n{ 0 }, sum{ 0 }, maximum{0};

addValueToVar(n);

vector <int> myVector(n, 0);

addValuesToVector(myVector,n);

printVector(myVector);

for (size\_t i = 0; i < n; i++)

{

sum = sum + myVector.at(i);

maximum = findMaxVar(myVector.at(i), maximum);

}

if (maximum < (sum - maximum))

{

cout << "DA";

}

else

{

cout << "NU";

}

return 0;

}

1. Se dă un număr n. Afișați rezultatul operației 2^n.

Solutie:

int main()

{

int n{ 0 }, produs{ 1 };

cin >> n;

for (size\_t i = 0; i < n; i++)

{

produs = produs \* n;

}

cout << "Produsul este: " << produs;

return 0;

}

1. Se citesc n numere naturale. Determinați pentru fiecare dintre ele dacă este par sau impar

Solutie:

void printVector(vector <int> yourVector)

{

cout << endl;

cout << "Your vector: " << endl;

cout << "[";

for (size\_t i = 0; i < yourVector.size(); i++)

{

cout << yourVector.at(i) << ",";

}

cout << "]";

cout << endl;

}

void addValuesToVector(vector <int>& yourVector, int vectorSize)

{

for (size\_t i = 0; i < vectorSize; i++)

{

cout << "Value at index " << i << ": ";

cin >> yourVector.at(i);

}

}

int main()

{

int n{ 0 };

cin >> n;

vector <int> myVector(n, 0);

addValuesToVector(myVector, n);

printVector(myVector);

cout << endl;

for (size\_t i = 0; i < myVector.size(); i++)

{

if (myVector.at(i) % 2 == 1)

{

cout << "1" << endl;

}

else

{

cout << "0" << endl;

}

}

return 0;

}