Selecția proporțională

Scrieți un program care sa aplice o selecție proporțională (sau Monte Carlo sau Softmax) pentru a alege din o mulțime de (maxim 30) opțiuni, fiecare cu evaluarea sa dată.

Descrierea sa (cu exemplu) o găsiți la http://inf.ucv.ro/documents/cstoean/c7IA_27.pdf, slide-urile 31-35.

Iată un exemplu de intrare a problemei:
30 10 5 60 90 20

Observații suplimentare:

- Setați numărul de opțiuni ca și constantă în program;
- Introduceți evaluările pentru fiecare acțiune în parte ca și valori direct în program, nu le citiți de la tastatură;
- Afișați la final ce acțiune a fost aleasă (indexul său) și care a fost evaluarea sa.

Soluții

Soluția 1

Observații: Am adus câteva modificări la programul primit ca temă. Afișează valorile pentru p, q și r, astfel încât să se poată înțelege de ce s-a ales opțiunea respectivă.

```
//
// main.cpp
// LS015
//
// Created by Teea Eliade on 25/05/2020.
// Copyright © 2020 Teea Eliade. All rights reserved.
//
#include <iostream>
#include <time.h>
#define NR_OPTIUNI 6
#define N 30
using namespace std;
int main()
{
	double p[N] = { 30,10,5,60,90,20 }, q[N] = {}, r = 0;
	int sum = 0;//pentru q
	srand(time(NULL));
	/*//Daca se ruleaza programul de multe ori imediat, primul numar generat este similar.
	Din acest motiv apelam o data rand(), fara sa folosim valoarea.
	De la cea de a doua apelare, rand() produce valori distincte. */
	rand();
```

```
	for (int i = 0; i < NR_OPTIUNI; i++)
	{
		sum += p[i];
		for (int j = 0; j <= i; j++)
			q[i] += p[j];
	}
	for (int i = 0; i < NR_OPTIUNI; i++)
		q[i] /= sum;

	cout << "Optiunile sunt:" << endl;
	for (int i = 0; i < NR_OPTIUNI; i++)
		cout << "p[" << i << "] = " << p[i] << endl;

	cout << endl << "Vectorul calculat q este:" << endl;
	for (int i = 0; i < NR_OPTIUNI; i++)
		cout << "q[" << i << "] = " << q[i] << endl;
	r = ((double)rand() / (RAND_MAX));
	cout << endl << "r = " << r << endl;
	int i = 0;
	while (q[i] < r)
		i++;

	cout << endl << "Optiunea selectata are indexul " << i << " si se refera la
evaluarea p " << p[i] << endl;
}
```

Programul rulat

```
Optiunile sunt:
p[0] = 30
p[1] = 10
p[2] = 5
p[3] = 60
p[4] = 90
p[5] = 20

Vectorul calculat q este:
q[0] = 0.139535
q[1] = 0.186047
q[2] = 0.209302
q[3] = 0.488372
q[4] = 0.906977
q[5] = 1

r = 0.311869

Optiunea selectata are indexul 3 si se refera la evaluarea p 60
```

Soluția 2

Observații: Și aici am făcut câteva mici modificări – l-am afișat pe r, am mai apelat o dată „`rand()`” și am introdus constanta „count” cu „#define” fiindcă aveam eroare cu „const”. „#define” funcționează la fel în C și C++.

```
/**
 * L15. Scrieti un program care sa aplice o selectie proportionala (sau Monte
 * Carlo sau Softmax) pentru a alege din o multime de (maxim 30) optiuni,
 * fiecare cu evaluarea sa data.
 * Descrierea sa (cu exemplu) o gasiti la
 * http://inf.ucv.ro/documents/cstoean/c7IA_27.pdf, slide-urile 31-35.
```

```
 * Iata un exemplu de intrare a problemei: 30 10 5 60 90 20
 * Observatii suplimentare:
 * - Setati numarul de optiuni ca si constanta in program;
 * - Introduceti evaluarile pentru fiecare actiune in parte ca si valori direct
 * in program, nu le cititi de la tastatura;
 * - Afisati la final ce actiune a fost aleasa (indexul sau) si care a fost
 * evaluarea sa.
 */
#include <cstdlib>   // RAND_MAX, srand(), rand()
#include <ctime>     // time()
#include <iostream>  // cout, cin

int options[] = { 2, 5, 17, 90, 32, 29, 56, 72, 84, 49 };
#define count sizeof(options) / sizeof(options[0])

using namespace std;

float q[count] = { 0 };

/**
 * @brief Genereaza un numar real intre limitele date
 * @param minimum
 * @param maximum
 * @result numarul generat
 *
 * https://stackoverflow.com/questions/686353/random-float-number-generation#686373
 * Probabil ar trebui folosite metode standard in C++ (v.
 * https://isocpp.org/files/papers/n3551.pdf)
 */
float random(float minimum = 0, float maximum = 1) {
    return minimum + rand() * (maximum - minimum) / RAND_MAX;
}

/**
 * @brief Calculeaza valorile vectorului global q, cf. algoritmului din curs,
 * pag. 33
 */
void compute_q() {
    float sum = 0;
    for (int i = 0; i < count; i++) {
        sum += options[i];
    }

    float temp = 0;
    for (int i = 0; i < count; i++) {
        temp += options[i];
        q[i] = temp / sum;
    }
}

/**
 * @brief Selectie Monte Carlo, cf. algoritmului din curs, pag. 35
 * @return indexul din options selectat
 */
int roulette() {
    float r = random();
    cout << endl << endl << "\tr = " << r << endl;
    int i = 0;
    while (q[i] < r) {
        ++i;
    }
    return i;
```

```
}

/**
 * @brief Afiseaza o lista de obiecte de tip oarecare (T)
 * @param array
 */
template <typename T>
void print_array(T* array) {
    cout << "{";
    for (int i = 0; i < count - 1; i++) {
        cout << array[i] << ", ";
    }
    cout << array[count - 1] << "}";
}

/**
 * @brief Inceputul executiei
 */
int main() {
    srand(time(0));
    rand();
    cout << "Optiunile sunt:\n\toptions[" << count << "] = ";
    print_array(options);

    compute_q();
    cout << "\n\n\tQ[" << count << "] = ";
    print_array(q);

    int counter = 1;
    while (true) {
        int result = roulette();
        cout << "\n\nRezultatul selectiei nr. " << counter << ":\n\toptions["
            << result << "] = " << options[result] << " (q = " << q[result]
            << ")";

        cout << "\n\nContinuam? (y/n) ";
        char answer;
        cin >> answer;
        if (answer != 'y') {
            break;
        }
        counter++;
    }
}
```

Programul rulat

```
Optiunile sunt:
        options[10] = {2, 5, 17, 90, 32, 29, 56, 72, 84, 49}

        Q[10] = {0.00458716, 0.016055, 0.0550459, 0.261468, 0.334862, 0.401376,
0.529817, 0.694954, 0.887615, 1}

        r = 0.763695


Rezultatul selectiei nr. 1:
        options[8] = 84 (q = 0.887615)

Continuam? (y/n)
```