**PREGUNTA 1**

#include <iostream>

#include <vector>

#include <string>

using *namespace* std;

*int* main()

{

    srand(time(NULL));

    string nombre\_torneo;

    getline(cin, nombre\_torneo);

    vector<string> jugadores;

    for (*int* i = 0; i < 2; i++)

    {

        string nombre;

        getline(cin, nombre);

        jugadores.push\_back(nombre);

    }

*int* n;

    do

    {

        cin >> n;

    } while (n < 5 || n > 15);

    vector<*int*> puntajes = {0, 3, 10};

    vector<pair<string, *int*>> encuentro\_iz;

    vector<pair<string, *int*>> encuentro\_der;

    for (*int* i = 0; i < n; i++)

    {

*int* random\_nombre = rand() % 2 + 1;

*int* random\_puntaje = rand() % 3;

*int* random\_puntaje2;

        if (random\_puntaje == 1)

            random\_puntaje2 = 1;

        else if (random\_puntaje == 0)

            random\_puntaje2 = 2;

        else

            random\_puntaje2 = 0;

        pair<string, *int*> jugador\_puntaje\_iz = {jugadores[random\_nombre - 1], puntajes[random\_puntaje]};

        pair<string, *int*> jugador\_puntaje\_der = {jugadores[2 - random\_nombre], puntajes[random\_puntaje2]};

        encuentro\_iz.push\_back(jugador\_puntaje\_iz);

        encuentro\_der.push\_back(jugador\_puntaje\_der);

    }

*int* puntaje\_total\_1 = 0;

*int* puntaje\_total\_2 = 0;

*int* j1\_encuentros\_ganados = 0;

*int* encuentros\_empatados = 0;

*int* j2\_encuentros\_perdidos = 0;

    for (*int* i = 0; i < encuentro\_iz.size(); i++)

    {

        if (encuentro\_iz[i].first == jugadores[0])

        {

            puntaje\_total\_1 += encuentro\_iz[i].second;

            puntaje\_total\_2 += encuentro\_der[i].second;

        }

        else

        {

            puntaje\_total\_1 += encuentro\_der[i].second;

            puntaje\_total\_2 += encuentro\_iz[i].second;

        }

        if (encuentro\_iz[i].first == jugadores[0] && encuentro\_iz[i].second > encuentro\_der[i].second)

        {

            j1\_encuentros\_ganados++;

        }

        else if (encuentro\_der[i].first == jugadores[0] && encuentro\_der[i].second > encuentro\_iz[i].second)

        {

            j1\_encuentros\_ganados++;

        }

        if (encuentro\_iz[i].second == encuentro\_der[i].second)

        {

            encuentros\_empatados++;

        }

        if (encuentro\_iz[i].first == jugadores[1] && encuentro\_iz[i].second < encuentro\_der[i].second)

        {

            j2\_encuentros\_perdidos++;

        }

        else if (encuentro\_der[i].first == jugadores[1] && encuentro\_der[i].second < encuentro\_iz[i].second)

        {

            j2\_encuentros\_perdidos++;

        }

        cout << encuentro\_iz[i].first << "-" << encuentro\_iz[i].second << "#" << encuentro\_der[i].first << "-" << encuentro\_der[i].second << endl;

    }

    cout << endl;

    cout << nombre\_torneo << endl;

    if (puntaje\_total\_1 > puntaje\_total\_2)

    {

        cout << "1. " << jugadores[0] << ", " << puntaje\_total\_1 << " puntos" << endl;

        cout << "2. " << jugadores[1] << ", " << puntaje\_total\_2 << " puntos" << endl;

    }

    else if (puntaje\_total\_2 > puntaje\_total\_1)

    {

        cout << "1. " << jugadores[1] << ", " << puntaje\_total\_2 << " puntos" << endl;

        cout << "2. " << jugadores[0] << ", " << puntaje\_total\_1 << " puntos" << endl;

    }

    else

    {

        cout << "1 y 2. Los dos jugadores empataron en puntajes" << endl;

    }

    cout << "3. " << j1\_encuentros\_ganados << " encuentros ganados" << endl;

    cout << "4. " << encuentros\_empatados << " encuentros empatados" << endl;

    cout << "5. " << j2\_encuentros\_perdidos << " encuentros perdidos" << endl;

    return 0;

}

**PREGUNTA 2:**

#include <iostream>

#include <vector>

using *namespace* std;

vector<*int*> KMPpreprocess(string *pattern*)

{

*int* m = *pattern*.length();

    vector<*int*> pi(m);

    pi[0] = 0;

*int* j = 0;

    for (*int* i = 1; i < m; i++)

    {

        while (j > 0 && *pattern*[j] != *pattern*[i])

        {

            j = pi[j - 1];

        }

        if (*pattern*[j] == *pattern*[i])

        {

            j++;

        }

        pi[i] = j;

    }

    return pi;

}

*void* KMPsearch(string *text*, string *pattern*, vector<*int*> &*cantidad*)

{

*int* cant\_apariciones = 0;

*int* n = *text*.length();

*int* m = *pattern*.length();

    vector<*int*> pi = KMPpreprocess(*pattern*);

*int* j = 0;

    vector<*int*> first\_pos;

    for (*int* i = 0; i < n; i++)

    {

        while (j > 0 && *pattern*[j] != *text*[i])

        {

            j = pi[j - 1];

        }

        if (*pattern*[j] == *text*[i])

        {

            j++;

        }

        if (j == m)

        {

            j = pi[j - 1];

            first\_pos.push\_back(i - m + 1);

            cant\_apariciones++;

        }

    }

    cout << "Cantidad de apariciones: " << cant\_apariciones << endl;

*cantidad*.push\_back(cant\_apariciones);

    for (*int* i = 0; i < first\_pos.size(); i++)

    {

        cout << "Indice: " << first\_pos[i] + 1 << endl;

    }

    if (cant\_apariciones > 5)

    {

        cout << "La cadena tiene varios patrones" << endl;

    }

    if (!first\_pos.empty())

    {

*text*.replace(first\_pos[0], 4, "BBBB");

*text*.replace(first\_pos[first\_pos.size() - 1], 4, "BBBB");

        cout << "Cadena modificada: " << endl;

        cout << *text* << endl;

    }

}

*int* main()

{

    srand(time(NULL));

*int* n;

    do

    {

        cout << "Ingresar numero de simulaciones:" << endl;

        cin >> n;

    } while (n <= 0 || n >= 100);

    vector<*int*> cant\_genes\_totales;

    while (n--)

    {

*int* c;

        do

        {

            cout << "Ingresar cantidad de elementos en la cadena:" << endl;

            cin >> c;

        } while (c <= 50 || c >= 10000);

        vector<*char*> caracteres = {'G', 'A', 'H', 'M', 'O'};

        string cadena;

        for (*int* i = 0; i < c; i++)

        {

*int* random = rand() % 5;

            cadena.push\_back(caracteres[random]);

        }

        cout << "Cadena: " << endl;

        cout << cadena << endl;

        string gen;

        for (*int* i = 0; i < 4; i++)

        {

*int* random;

            if (i == 0)

            {

                do

                {

                    random = rand() % 5;

                } while (random != 1 && random != 4);

            }

            else

            {

                random = rand() % 5;

            }

            gen.push\_back(caracteres[random]);

        }

        cout << "Gen: " << gen << endl;

        KMPsearch(cadena, gen, cant\_genes\_totales);

        cout << endl;

    }

*int* total = 0;

*int* menor = INT\_MAX;

*int* pos;

    for (*auto* i : cant\_genes\_totales)

        cout << i << " ";

    cout << endl;

    for (*int* i = 0; i < cant\_genes\_totales.size(); i++)

    {

        total += cant\_genes\_totales[i];

        if (cant\_genes\_totales[i] < menor)

        {

            menor = cant\_genes\_totales[i];

            pos = i;

        }

    }

    cout << "Cantidad de veces que aparecieron los genes en las cadenas: " << total << endl;

    cout << "En la simulacion " << pos + 1 << " se encontraron menos genes.";

    return 0;

}