Работа призера заключительного этапа

командной инженерной олимпиады школьников

Олимпиада Национальной технологической инициативы

Профиль «Технологии беспроводных связей»

Кабдыкайыров Ержан Кабижанулы

Класс: 10
Школа: НИШ ФМН

Город: Алматы
Регион: Казахстан

Уникальный номер участника: 6033530
ID-степик: 36454746
Параллель: 10-11

Команда на заключительном этапе: EASY НТИ

Результаты заключительного этапа:

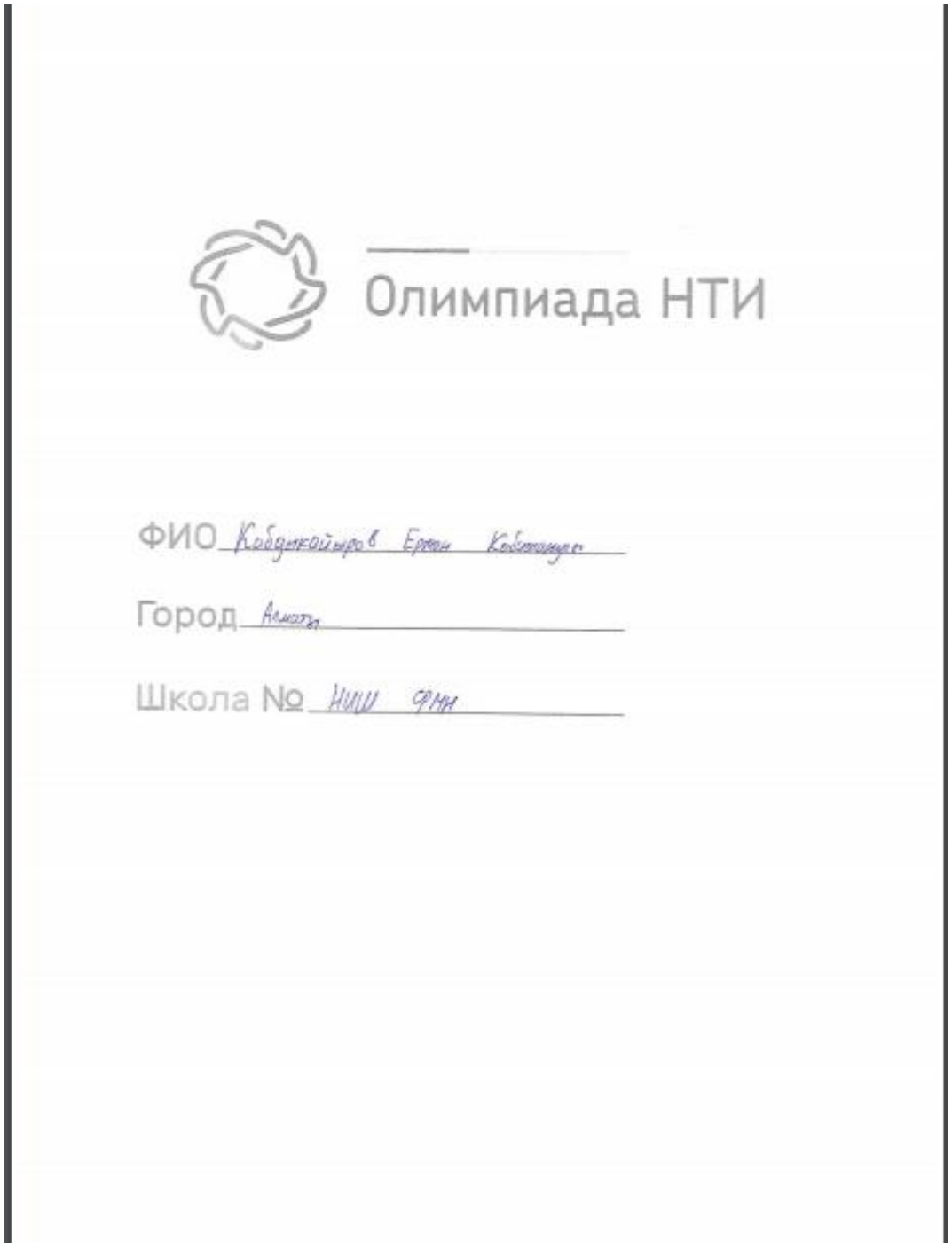
Результаты заключительного этапа (оценка по ненормированным баллам):

Индивидуальная часть (не норм)							Командная часть (не норм)							Результат, нормир. На 100 баллов
	Математика	Информатика			ИТОГО	Макс балл							Макс балл	
№	ИТОГО	1	2	3			1	2	3	4	5	ИТОГО		
	4	25	40	19,69	49,375	**100**	3	0	15,11	15	0	33,108	126	44,3375

Индивидуальная часть

МАТЕМАТИКА

Решение призёра:


Олимпиада НТИ

ФИО Кобдикайыров Ерлан Кайратович

Город Алматы

Школа № НИШ ФМН

Командная инженерная олимпиада «Олимпиада НТИ»

Направление ГТС

Предмет Математика

Номер участника 6033630

2.

а) $P(\text{Алиса отправит 0 (сокр. } A_0)) = p$

$P(\text{Алиса отправит 1 (сокр. } A_1)) = 1-p$

б)

$P(\text{Боб получит 0 (сокр. } Б_0)) = \underbrace{P(A_1)\cdot\varepsilon}_{\text{Алиса отправила 1, но случилась ошибка}} + \underbrace{P(A_0)\cdot(1-\varepsilon)}_{\text{Алиса отправила 0 и всё прошло без ошибок}} = p - 2\varepsilon p + \varepsilon$

$P(\text{Боб получит 1 (сокр. } Б_1)) = \underbrace{P(A_0)\cdot\varepsilon}_{\text{ошибка}} + \underbrace{P(A_1)\cdot(1-\varepsilon)}_{\text{прошло}} = 1 + 2\varepsilon p - \varepsilon - p$

$P(Б_0) + P(Б_1) = p - 2\varepsilon p + \varepsilon + 1 + 2\varepsilon p - \varepsilon - p = 1$ — вот так проверяется

или (в развернутом виде)

в)

$P(\text{Алиса отправила } A_1, \text{ Боб принял } Б_1) = (1-p)\cdot(1-\varepsilon) = 1 - \varepsilon - p + \varepsilon p$ (Алиса отправила 1, Боб принял 1)

$P(\text{Алиса отправила } A_1, \text{ Боб принял } Б_0) = (1-p)\cdot\varepsilon = \varepsilon - \varepsilon p$ (Алиса отправила 1, Боб принял 0)

$P(\text{Алиса отправила } A_0, \text{ Боб принял } Б_1) = p\cdot(1-\varepsilon) = p - \varepsilon p$ (Алиса отправила 0, Боб принял 1)

$P(\text{Алиса отправила } A_0, \text{ Боб принял } Б_0) = p - \varepsilon$ (Алиса отправила 0, Боб принял 0)

г) Данные совпадут, если бит, отправленный Алисой совпадет с битом, принятым Бобом.

В этом случае:

$P(\text{совпадения}) = P(A_1 Б_1) + P(A_0 Б_0) = 1 - \varepsilon - p + \varepsilon p + p - \varepsilon p = 1 - \varepsilon$

Пропускная способность канала прямо пропорциональна $P(\text{совпадения})$, из-за чего макс. $R = 1$ в случае если $\varepsilon = 0$ и мин. $R = 0$ если $\varepsilon = 1$.

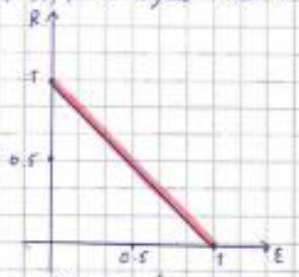


График зависимости между ε и R

Балл призёра за математику: 4

ИНФОРМАТИКА

Задача 1. Платежные каналы

Решение призёра:

```
#include <iostream>
 #include <vector>
 using namespace std;
 const int NMAX = 100001;
 vector<int> g[NMAX];
 bool used[NMAX];
 void dfs(int u)
{
        used[u] = true;
        for (int v : g[u])
       if (!used[v])
              dfs(v);
}
 int main()
{
       ios_base::sync_with_stdio(false);
       cin.tie(nullptr);
       int n, m;
        cin >> n >> m;
       int a, b;
        while (m--)
       {
        cin >> a >> b;
        g[a].push_back(b);
       }
       int s, t;
        cin >> s >> t;
        dfs(s);
        cout << (used[t] ? "Yes" : "No");
}
```

```
#include <bits/stdc++.h>
 using namespace std;
 vector<int> g[100000];
 bool u[100000];
 void dfs(int x) {
    u[x] = true;
    for(int i = 0; i < g[x].size(); i++) {
        int to = g[x][i];
        if(!u[to]) {
            dfs(to);
        }
    }
}
 int main()
{
```

```
    int n, m; cin >> n >> m;
    for(int i = 0; i < m; i++) {
        int ai, bi;
        cin >> ai >> bi;
        g[ai].push_back(bi);
    }
    int s, t;
    cin >> s >> t;
    dfs(s);
    if(u[t]) {
        cout << "Yes";
    }
    else {
        cout << "No";
    }
    return 0;
}
```

```
#include <bits/s tdc++.h>
 using namespace std;
 vector<int> g[300000];
 bool u[300000];
 void dfs(int x) {
    u[x] = true;
    for(int i = 0; i < g[x].size(); i++) {
        int to = g[x][i];
        if(!u[to]) {
            dfs(to);
        }
    }
}
 int main()
{
    int n, m; cin >> n >> m;
    for(int i = 0; i < m; i++) {
        int ai, bi;
        cin >> ai >> bi;
        g[ai].push_back(bi);
    }
    int s, t;
    cin >> s >> t;
    dfs(s);
    if(u[t]) {
        cout << "Yes";
    }
    else {
        cout << "No";
    }
    return 0;
}
```

Полученный балл:25

Задача 2. Платежные каналы с комиссией

Решение призёра:

```
#include <bits/stdc++.h>
 using namespace std;
 int cost[300000];
 vector<pair<int, int> > graph[300000];
 bool u[300000];
 bool qu[300000];
 void dfs(int x) {
    u[x] = true;
    for(int i = 0; i < graph[x].size(); i++) {
        int to = graph[x][i].first;
        if(!u[to]) {
            dfs(to);
        }
    }
}
 int main()
{
    int n, m;
    cin >> n >> m;
    for(int i = 0; i < m; i++) {
        int a, b, c;
        cin >> a >> b >> c;
        graph[a].push_back(make_pair(b, c));
    }
    int s, t;
    cin >> s >> t;
    int q, channelCost;
    cin >> q >> channelCost;
    for(int i = 0; i < q; i++) {
        int a, b, c;
        cin >> a >> b >> c;
        graph[a].push_back(make_pair(b, channelCost + c));
    }
    dfs(s);
    if(!u[t]) {
        cout << -1;
        return 0;
    }
    for(int i = 0; i < 300000; i++) {
        cost[i] = (1 << 31) - 1;
    }
    cost[s] = 0;
    queue<pair<int, int> > que;
    que.push(make_pair(s, s));
    while(!que.empty()) {
        pair<int, int> cur = que.front();
        //cout << cur.first << " " << cur.second << "| ";
        que.pop();
        for(int i = 0; i < graph[cur.first].size(); i++) {
            int to = graph[cur.first][i].first;
```

```
            //cout << to << " ";
            if(!qu[to]) {
                qu[to] = true;
                que.push(make_pair(to, cur.first));
            }
            cost[to] = min(cost[to], cost[cur.first] + graph[cur.first][i].second);
        }
        //cout << endl;
    }
    cout << cost[t];
    return 0;
}


#include <bits/stdc++.h>
using namespace std;
 long long cost[300000];
vector<pair<int, long long> > graph[300000];
bool u[300000];
bool qu[300000];
void dfs(int x) {
    u[x] = true;
    for(int i = 0; i < graph[x].size(); i++) {
        int to = graph[x][i].first;
        if(!u[to]) {
            dfs(to);
        }
    }
}
int main()
{
    int n, m;
    cin >> n >> m;
    for(int i = 0; i < m; i++) {
        int a, b, c;
        cin >> a >> b >> c;
        graph[a].push_back(make_pair(b, c));
    }
    int s, t;
    cin >> s >> t;
    long long q, channelCost;
    cin >> q >> channelCost;
    for(int i = 0; i < q; i++) {
        int a, b, c;
        cin >> a >> b >> c;
        graph[a].push_back(make_pair(b, channelCost + c));
    }
    dfs(s);
    if(!u[t]) {
        cout << -1;
        return 0;
    }
    for(int i = 0; i < 300000; i++) {
        cost[i] = -1;
```

```
    }
    cost[s] = 0;
    queue<pair<int, int> > que;
    que.push(make_pair(s, s));
    while(!que.empty()) {
        pair<int, int> cur = que.front();
        //cout << cur.first << " " << cur.second << "| ";
        que.pop();
        for(int i = 0; i < graph[cur.first].size(); i++) {
            int to = graph[cur.first][i].first;
            //cout << to << " ";
            int calculatedCost = cost[cur.first] + graph[cur.first][i].second;
            if(cost[to] == -1) {
                cost[to] = calculatedCost;
            }
            else if(calculatedCost < cost[to]) {
                qu[to] = false;
                cost[to] = calculatedCost;
            }
            if(!qu[to]) {
                qu[to] = true;
                que.push(make_pair(to, cur.first));
            }
        }
        //cout << endl;
    }
    cout << cost[t];
    return 0;
}


#include <bits/stdc++.h>
using namespace std;
long long cost[300000];
vector<pair<int, long long> > graph[300000];
bool u[300000];
bool qu[300000];
void dfs(int x) {
    u[x] = true;
    for(int i = 0; i < graph[x].size(); i++) {
        int to = graph[x][i].first;
        if(!u[to]) {
            dfs(to);
        }
    }
}
int main()
{
    int n, m;
    cin >> n >> m;
    for(int i = 0; i < m; i++) {
        int a, b, c;
        cin >> a >> b >> c;
        graph[a].push_back(make_pair(b, c));
```

```
    }
    int s, t;
    cin >> s >> t;
    long long q, channelCost;
    cin >> q >> channelCost;
    for(int i = 0; i < q; i++) {
        int a, b, c;
        cin >> a >> b >> c;
        graph[a].push_back(make_pair(b, channelCost + c));
    }
    dfs(s);
    if(!u[t]) {
        cout << -1;
        return 0;
    }
    for(int i = 0; i < 300000; i++) {
        cost[i] = -1;
    }
    cost[s] = 0;
    queue<pair<int, int> > que;
    que.push(make_pair(s, s));
    while(!que.empty()) {
        pair<int, int> cur = que.front();
        //cout << cur.first << " " << cur.second << "| ";
        que.pop();
        for(int i = 0; i < graph[cur.first].size(); i++) {
            int to = graph[cur.first][i].first;
            //cout << to << " ";
            int calculatedCost = cost[cur.first] + graph[cur.first][i].second;
            if(cost[to] == -1) {
                cost[to] = calculatedCost;
            }
            else if(calculatedCost < cost[to]) {
                qu[to] = false;
                cost[to] = calculatedCost;
            }
            if(!qu[to]) {
                qu[to] = true;
                que.push(make_pair(to, cur.first));
            }
        }
        //cout << endl;
    }
    cout << cost[t];
    return 0;
}
```

Полученный балл: 40

Задача 3. Дерево транзакций

Решение призёра:

```
#include <bits/stdc++.h>
```

```
using namespace std;
vector<pair<int, int> > graph[300001];
//bool u[300001];
int cnt = 0;
long long req = 0;
void dfs(int x, long long s, int p = -1) {
    //u[x] = true;
    if(s == req) {
        cnt++;
    }
    for(int i = 0; i < graph[x].size(); i++) {
        int to = graph[x][i].first;
        int val = graph[x][i].second;
        if(to != p) {
            dfs(to, s ^ val, x);
        }
    }
}
int main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(NULL);
    cout.tie(NULL);
    int n, q;
    cin >> n >> q;
    for(int i = 0; i < n - 1; i++) {
        int a, b, v;
        cin >> a >> b >> v;
        graph[a].push_back(make_pair(b, v));
        graph[b].push_back(make_pair(a, v));
    }
    for(int i = 0; i < q; i++) {
        cnt = 0;
        int vi; cin >> vi;
        cin >> req;
        dfs(vi, 0);
        cout << cnt << endl;
    }
    return 0;
}
```

Полученный балл: 19,69

Командная часть

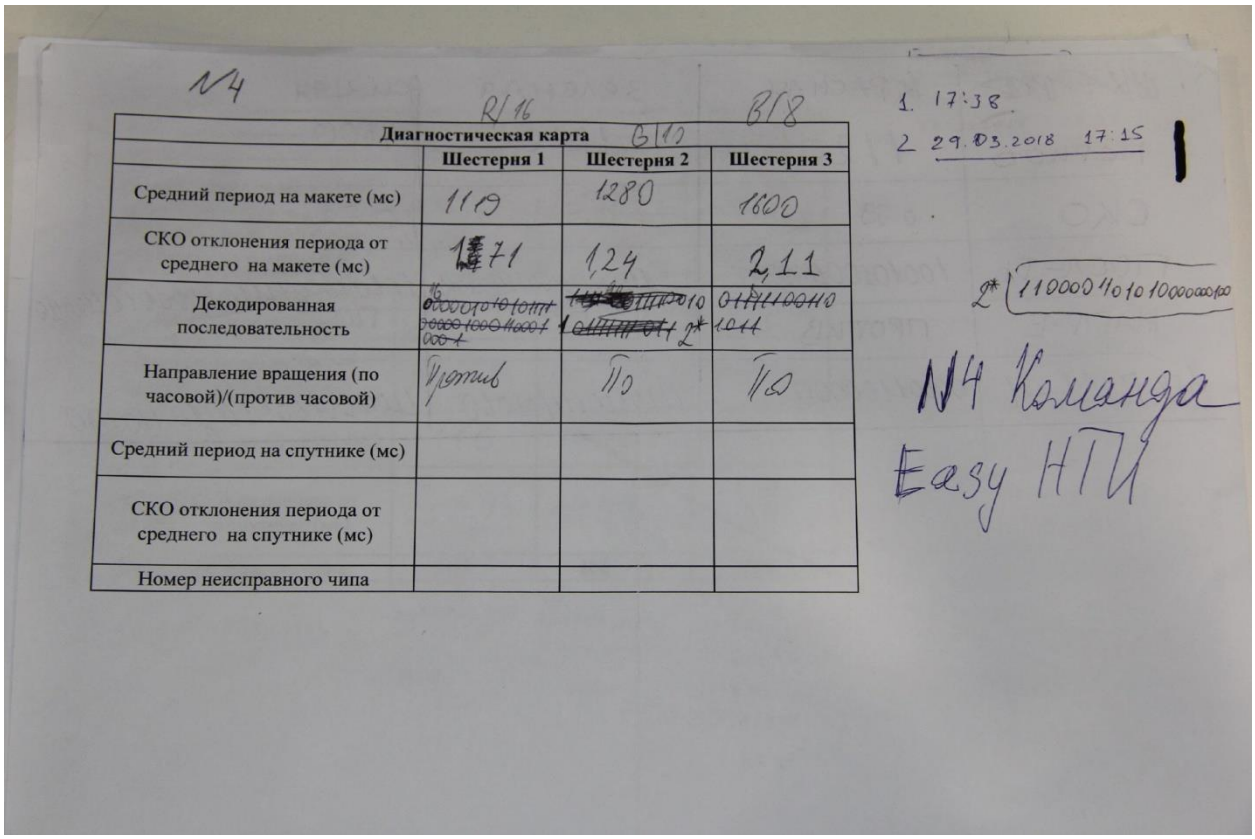
Результаты были получены в рамках выступления команды: EASY НТИ

Личный состав команды:

- Кабдыкайыров Ержан Кабижанулы
- Смагулов Ансар Серикович
- Акижанов Данияр Тимурович
- Умирбаев Алихан Талгатович

Задача 1.

Решение:

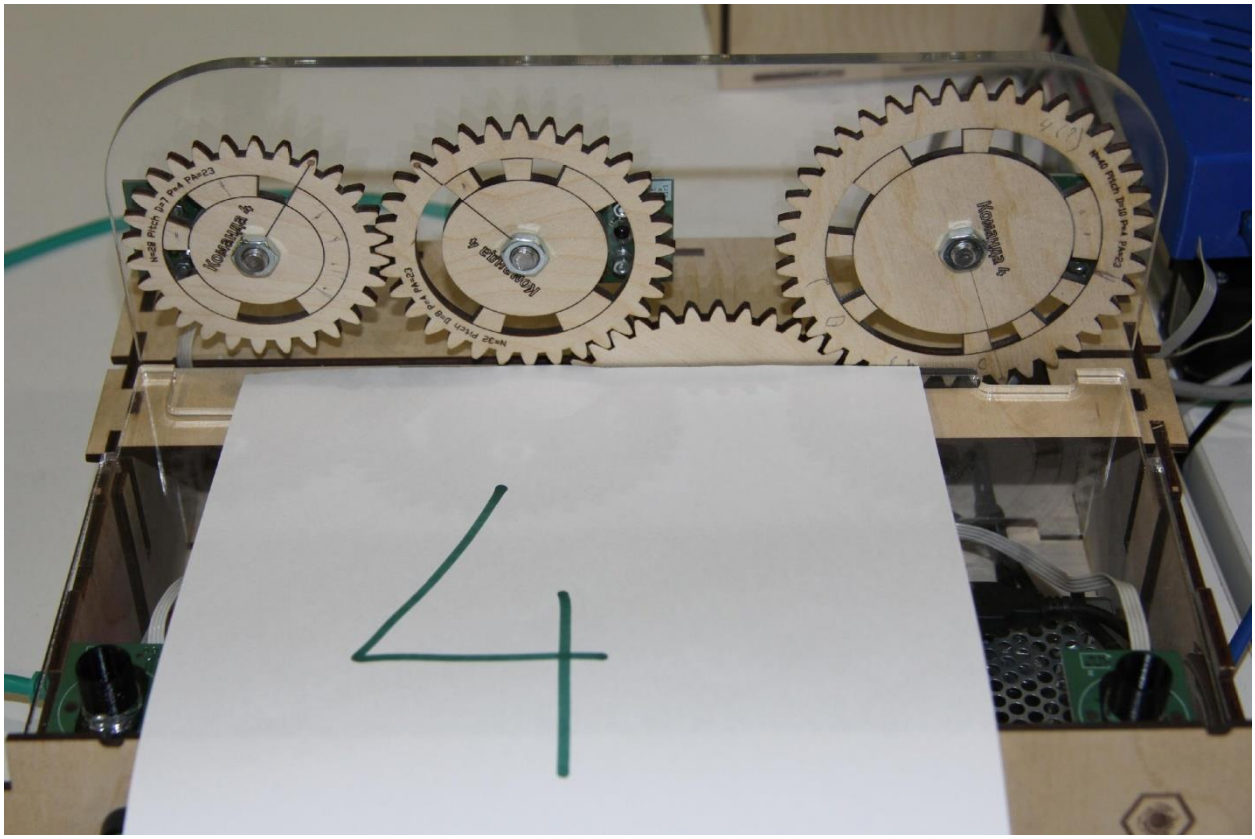


N4

Диагностическая карта	Шестерня 1	Шестерня 2	Шестерня 3
Средний период на макете (мс)	1109	1280	1600
СКО отклонения периода от среднего на макете (мс)	1,71	1,24	2,11
Декодированная последовательность			
Направление вращения (по часовой)/(против часовой)	Против	По	По
Средний период на спутнике (мс)			
СКО отклонения периода от среднего на спутнике (мс)			
Номер неисправного чипа			

1. 17:38
2. 29.03.2018 17:15

N4 Команда Easy HTU



Полученный результат: 3 балла

Задача 2.

Командой не была решена, 0 баллов

Задача 3

Решение задачи – программный код.

Решение:

```
#include <stdio.h>
#include <signal.h>
#include <unistd.h>
#include <math.h>
#include <fcntl.h>
#include "shmipc.h"
#include "client.h"

int clampSpeed(int speed) {
    if(abs(speed) >= 95) {
        return 95;
    }

    return abs(speed);
}

double getDxMultiplier(int dx) {
    if(dx > 30) {
        return -1.0;
    }
    else if(dx < -30) {
            return 1.0;
    }

    return (double)dx / 30.0;
}
int main(int argc, char *argv[])
{
    Client control;
    int curSpeed;
    bool direction;
    int dx;
    int prevDx = 0;
    int iter = 0;
    control.start();
    control.right(60);

    int delayTime = 10;

    double kp = 0.7;
    double kd = 0.0;

    int movingDir = 1;
```

```
    //printf("Client initialized");

    while(true) {
        dx = control.getDx();

        printf("dx: %d, speed: %d\n", dx, curSpeed);

        if(abs(dx) <= 1000){
            curSpeed = round((double)dx * kp);

            direction = (curSpeed < 0);
            curSpeed = round(pow((double)clampSpeed(curSpeed) / 95.0, 0.5) * 95.0);
            //if(clampSpeed(curSpeed) < 30) {
            //    curSpeed = round((double)curSpeed * pow((double)dx / 300.0, 2.0));
            //})
            int dxDiff = dx - prevDx;

            if(direction) {
                control.right(clampSpeed(curSpeed));
                movingDir = 1;
            }
            else {
                control.left(clampSpeed(curSpeed));
                movingDir = -1;
            }
        }
        else{
            if(movingDir == 1) {
                control.right(85);
            }
            else if(movingDir == -1) {
                control.left(85);
            }
            if(control.ifPositionLeft()) {
                movingDir = 1;
            }
            else if(control.ifPositionRight()) {
                movingDir = -1;
            }
        }

        iter++;
        usleep(1000 * delayTime);
    }

}
```

Полученный результат: реализовано автоматическое наведение «радара» на «спутник»

Полученный результат: 15,108 балла

Задача 4

Решение: решение представляет собой текстовый файл с цифровым образом шестерни. Результирующий файл расположен на (https://drive.google.com/drive/u/0/folders/1SxB7e_3RsQoz_1-j0GP3ca4LgLC3WnzK), команда «Easy NTI»

Полученный результат: получено изображение шестерни с требуемой последовательностью прорезей и стенок, полученная модель успешно напечатана на 3D принтере.

Полученный результат: 15 баллов

Задача 5

Командой не была решена, 0 баллов