olimp.tech

Олимпиада
«Технологическое
предпринимательство»

Наименование образовательного учреждения: ФГБОУ ВО «Сибирский государственный аэрокосмический университет имени академика М.Ф. Решетнева»

Очный этап Олимпиады «Технологическое предпринимательство»

Бланк заполняется печатными буквами

Олимпиада по ИНФОРМАЦИОННЫМ ТЕХНОЛОГИЯМ класс 9

Фамилия КАЗАКОВ

Имя МИХАИЛ

Отчество ВЯЧЕСЛАВОВИЧ

Дата рождения 21.07.2001

Страна РФ

Регион КРАСНОЯРСКИЙ КРАЙ

Заполняется организатором

_____

olimp.tech

Задание №1

Несложно заметить, что символ ↑ представляет из себя единицу слева и зеркально отображенную единицу с правой стороны. Символ 2 представляет из себя двойку по схеме выше, символ 4 – четверку, символ 7 – семерку.

На основе рассуждений выше запишем все оставшиеся цифры:

3; 5; 6; 8; 9; 0.

Эти символы и будут недостающими символами этого алфавита.

Ответ: 3; 5; 6; 8; 9; 0.

olimp.tech

Задание №2

Программный код ниже представлен на языке программирования Python.

```
sum= 100
print(sum) # т.к. N≥1
n = int(input())
for _ in range(n-1):
    sum *= 2
    must_zero= len(str(sum)) //2 + len(str(sum)) %2 #кол-во
                                                    # нулей в числе(необход)
    if sum %(10** must_zero)≥ 5*(10**(must_zero-1)):
        sum += 10** must_zero # округление числа вверх
    sum //= 10** must_zero
    ~~print(sum)~~
    sum *= 10** must_zero
    print(sum)
```

olimp.tech

Страница 1

Задание №3

Программа представлена на языке Python.
Т.к. ограничений по времени не указано, то найдем все возможные пути робота, отберем те, что ведут к месту назначения и выберем кратчайший.
Также я считаю, что 1-й пример некорректен, т.к. робот может добраться таким способом:

```
S. # . .
# ↑ # . .
R→ . . . .
```

Здесь ответ будет 4

```
h,w = [int(x) for x in input.split()]
map = []
for i in range(h):
    line = input()
    map.append(list(line))
    if "R" in line:
        r_ind = (i, line.find("R"))
ends = [r_ind]
new_ends = []
paths = {r_ind: ends}
was_here = set(ends)
~~for end in~~
while True:
    for end in ends:
        for pos in [(0,-1), (0,1), (1,0), (-1,0)]:
            pos = (end[0] + pos[0], end[1] + pos[1])
            if map[pos[0]][pos[1]] == "#":
                continue
            if pos[0] < 0 or pos[0] ≥ w:
                continue
            if pos[1] < 0 or pos[1] ≥ h:
                continue
            if pos in was_here:
                continue
```

olimp.tech

Страница 2

Задание №3 (продолжение)

```
            paths[pos] = [pos] + paths[end]
            if map[pos[0]][pos[1]] == "S":
                print(len(paths[pos]))
                raise SystemExit
            was_here.add(pos)
            new_ends.append(pos)
    if not new_ends:
        print(-1)
        raise SystemExit
    ends = [i for i in new_ends]
    new_ends = []
```

Решение верно. В 1 примере действительно была ошибка.
Задача может быть оптимизирована поиском пути на графах.

olimp.tech

Страница 1

Задание №4

Программный код ниже приведен на языке Python.

```
x, n=[int(i) for i in input.split()]
s, w=[int(i) for i in input.split()]
q, m=[int(i) for i in input.split()]
sum=0
speed=0
spd_now=0
comps=0
progs=0
while True:
    if spd_now <=0: # устанавливаем баланс между
        if sum+s>x: # печатанием кода и его проверкой
            break
        else:
            progs+=1
            sum+=s
            if spd_now + w <=0:
                speed+=w
            else:
                speed+=abs(spd_now)
            spd_now+=w
    else:
        if sum+q>x:
            break
        else:
            comps+=1
            sum+=q
            if spd_now-m>=0:
                speed+=m
            else:
                speed+=spd_now
            spd_now-=m
    if speed>=n:
        break
# Теперь оптимизируем задачу по расходам
days=n//speed
if n%speed>0:
    days+=1
```

Задание №4 (продолжение)

```
if progs==0 or comps==0:
    print("Неверные данные введены")
tmp_days=0
tmp=speed
while True:
    if spd_now>=0:
        if spd_now - w≤0:
            tmp -= spd_now
        if tmp==0: break
        if n % tmp>0:
            tmp_days += 1 + n // tmp
        else:
            tmp_days = n // tmp
        if tmp_days==days:
            progs-=1
            sum-=s
        else:
            break
    else:
        if spd_now + m ≥0:
            tmp-=abs(spd_now)
        else:
            tmp=m
        if tmp==0:
            break
        if n % tmp>0:
            tmp_days += 1+ n//tmp
        else:
            tmp_days = n//tmp
        if tmp_days==days:
            comps-=1
            sum-=q
        else: break
print(sum)
print(progs, comps)
print(days)
```