

ТЕМА: ВВЕДЕНИЕ В ЯЗЫК ПРОГРАММИРОВАНИЯ «C++»

Домашнее задание 1

ЗАДАНИЕ 1

Пользователь вводит с клавиатуры символ. Определить, какой это символ: Буква, цифра, знак препинания или другое.

Подсказка 1

Если у Вас возникли сложности с выполнением данного задания, нажмите кнопку «Подсказка». С полным решением задания вы сможете ознакомиться, нажав на кнопку «Решение»

Решение 1

ЗАДАНИЕ 2

Написать программу подсчета стоимости разговора для разных мобильных операторов. Пользователь вводит длительность разговора и выбирает с какого на какой оператор он звонит. Вывести стоимость на экран.

Подсказка 2

Если у Вас возникли сложности с выполнением данного задания, нажмите кнопку «Подсказка». С полным решением задания вы сможете ознакомиться, нажав на кнопку «Решение»

Решение 2

ЗАДАНИЕ 3

Вася работает программистом и получает 50\$ за каждые 100 строк кода. За каждое третье опоздание на работу Васю штрафуют на 20\$. Реализовать меню:

- пользователь вводит желаемый доход Васи и количество опозданий, посчитать, сколько строк кода ему надо написать;
- пользователь вводит количество строк кода, написанное Васей и желаемый объем зарплаты. Посчитать, сколько раз Вася может опоздать;

- пользователь вводит количество строк кода и количество опозданий, определить, сколько денег заплатят Васе и заплатят ли вообще.

Подсказка 3

Если у Вас возникли сложности с выполнением данного задания, нажмите кнопку «Подсказка». С полным решением задания вы сможете ознакомиться, нажав на кнопку «Решение»

Решение 3

ПОДСКАЗКА К ЗАДАНИЮ 1

1. Какой тип данных используется для работы с отдельными символами?
2. Каждый символ имеет свой код. Чтобы узнать коды нужных символов необходима таблица ASCII-символов (американский стандартный код для обмена информацией, который определяет способ представления символов английского языка в виде чисел от 0 до 127).
3. Каким образом можно реализовать преобразование символьного типа в целочисленный (для получения ASCII-кода символа)?
4. Каким оператором условного ветвления реализуется ситуация, когда нужно последовательно проверить несколько условий (выполниться может только одно из них: символ либо буква, либо цифра, либо знак препинания) и предусмотреть альтернативный шаг в случае, когда они не выполняются?
5. Если необходимо проверить, находится ли число в определенном диапазоне (от — до), то какой логический оператор необходимо использовать в условии?
6. Диапазон кодов у строчных символов один, а у заглавных — другой. Какой логический оператор необходимо использовать в условии, если символ является буквой, когда он принадлежит любому из двух диапазонов кодов?

ПОДСКАЗКА К ЗАДАНИЮ 2

1. Звонки внутри сети (когда мобильный оператор, с которого звонят, и мобильный оператор, на который звонят, совпадают) бесплатны.
2. Нужно установить стоимость одной минуты звонка за пределы сети (на другие операторы) для каждого из предусмотренных в программе операторов.
3. После определения, с какого оператора поступает звонок, нет необходимости проверять, на какой оператор поступает звонок, если операторы разные, так как стоимость одной минуты звонка за пределы сети не зависит от того, на какой оператор звонят.
4. Если известна стоимость одной минуты звонка и длительность разговора в минутах, как вычислить стоимость разговора?
5. Каким оператором условного ветвления реализуется ситуация, когда нужно последовательно проверить несколько условий (выполниться может только одно из них:) и предусмотреть альтернативный шаг в случае, когда они не выполняются (пользователь ввел непредусмотренный мобильный оператор)?

ПОДСКАЗКА К ЗАДАНИЮ 3

1. Если известна стоимость 100 строк кода, как определить стоимость одной строчки кода?
2. Если известна стоимость одной строчки кода и желаемый доход, как определить, сколько строк кода надо написать?
3. Если известно общее количество опозданий и то, что штрафуются каждое третье опоздание, то как узнать, сколько раз будет применен штраф?
4. Если известно, сколько раз применяется штраф и стоимость одного штрафа, то как определить общую штрафную сумму?
5. Если количество опозданий от трех и более, то желаемый доход уменьшается на сумму штрафа, значит, к количеству строк кода, обеспечивающих желаемый доход (без учета штрафов) нужно добавить количество строк кода, которые покроют штрафную сумму.
6. Чтобы определить количество возможных опозданий на основании известного количества строк кода и желаемого объема зарплаты, необходимо найти вначале возможно допустимую сумму штрафов.
7. Для определения возможной зарплаты при известном количестве строк кода и числе опозданий, необходимо найти разницу между зарплатой за написанный код и суммой штрафов.

РЕШЕНИЕ ЗАДАНИЯ 1

Описание решения

После ввода символа пользователем выполняем его преобразования из символьного типа в целочисленный для получения ASCII-кода (<https://uk.wikipedia.org/wiki/ASCII>) введенного пользователем символа.

В примере ниже показан фрагмент ASCII-таблицы.

Бінарне	Дес.	Шіст.	Графічне	Бінарне	Дес.	Шіст.	Графічне	Бінарне	Дес.	Шіст.	Графічне
0010 0000	32	20	пропуск (sp)	0100 0000	64	40	@	0110 0000	96	60	`
0010 0001	33	21	!	0100 0001	65	41	A	0110 0001	97	61	a
0010 0010	34	22	"	0100 0010	66	42	B	0110 0010	98	62	b

Рисунок 1

Например, по данной таблице можно видеть, что ASCII-код символа «A» равен 65, а ASCII-код символа «a» равен 97. Аналогично можно найти ASCII-код символ «!», это будет 33.

Далее организуем последовательность условий, проверяющих, принадлежит ли код символа к одному из диапазонов кодов букв (два отдельных, непересекающихся диапазона для строчных и заглавных букв) или код принадлежит к диапазону цифр. Коды знаков препинания не лежат в одном диапазоне, поэтому необходимо проверить равенство кода введенного символа с кодами знаков препинания (!, - . : ; ?). Иначе — введенный символ относится к категории «другое».

Решение

1. Выводим строку в консоль с описанием названия программы (по желанию, данная строка кода может быть пропущена),

создаем необходимое число переменных — одну символьного типа для хранения введенного пользователем символа, другую целочисленную для хранения кода символа.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #5.1.1\n\n";
    int  ch_code;
    char ch;

    return 0;
}
```

2. Выводим в консоль строку с приглашением ввести символ, считываем его в переменную и преобразуем в целочисленный тип для получения кода символа. Явное преобразование типов данных выполняется с помощью оператора `()`. Внутри круглых скобок мы пишем тип (`int` в нашем случае, т.к. нам необходимо получить целочисленный ASCII-код символа).

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #5.1.1\n\n";
    int  ch_code;
```

```
char ch;  
cout << "Please, enter a character:\n";  
cin >> ch;  
ch_code = (int)ch;  
  
return 0;  
}
```

3. Далее проверяем, принадлежит ли код символа к одному из диапазонов кодов букв (два отдельных, непересекающихся диапазона для строчных, от 97 до 122 включительно, и заглавных букв, от 65 до 90 включительно).

Для реализации ситуации «от — до» используем логический оператор **И** (&&). Для реализации ситуации «код принадлежит к любому из диапазонов» используем логический оператор **ИЛИ** (||).

```
#include <iostream>  
using namespace std;  
  
int main()  
{  
    cout << "Home task #5.1.1\n\n";  
    int ch_code;  
    char ch;  
    cout << "Please, enter a character:\n";  
    cin >> ch;  
    ch_code = (int)ch;  
  
    if (((ch_code >= 65) && (ch_code <= 90)) ||  
        ((ch_code >= 97) && (ch_code <= 122)))
```



```
{  
    cout << "You entered a letter";  
}  
  
return 0;  
}
```

4. Согласно таблице ASCII-символов, коды цифр находятся в диапазоне от 48 до 57 включительно.

Так как выполниться может только одно из условий: символ либо буква, либо цифра, либо знак препинания, то для реализации процедуры последовательной проверки кода символа будем использовать оператор условного ветвления `if- else if - else`.

```
#include <iostream>  
using namespace std;  
  
int main()  
{  
    cout << "Home task #5.1.1\n\n";  
    int ch_code;  
    char ch;  
    cout << "Please, enter a character:\n";  
    cin >> ch;  
  
    ch_code = (int)ch;  
  
    if (((ch_code >= 65) && (ch_code <= 90)) ||  
        ((ch_code >= 97) && (ch_code <= 122)))  
    {
```

```
        cout << "You entered a letter";  
    }  
    else if ((ch_code >= 48) && (ch_code <= 57))  
    {  
        cout << "You entered a digit";  
    }  
  
    return 0;  
}
```

5. Коды знаков препинания не лежат в одном диапазоне, поэтому необходимо проверить равенство кода введенного символа с кодами следующих знаков препинания: !, - .; :? (знаки препинания , - . идут подряд от 44 до 46 включительно, можно организовать проверку на диапазон).

```
#include <iostream>  
using namespace std;  
  
int main()  
{  
    cout << "Home task #5.1.1\n\n";  
    int ch_code;  
    char ch;  
    cout << "Please, enter a character:\n";  
    cin >> ch;  
  
    ch_code = (int)ch;  
  
    if (((ch_code >= 65)&&(ch_code <= 90)) ||  
        ((ch_code >= 97) && (ch_code <= 122)))  
    {
```

```
        cout << "You entered a letter";
    }
    else if ((ch_code >= 48) && (ch_code <= 57))
    {
        cout << "You entered a digit";
    }
    else if ((ch_code == 33) || ((ch_code >= 44) &&
        (ch_code <= 46)) || (ch_code == 58) ||
        (ch_code == 59) || (ch_code == 63))
    {
        cout << "You entered a punctuation mark";
    }

    return 0;
}
```

6. Если условия, представленные в шагах 3-5, не выполняются, то введенный символ относится к категории «другое» (реализуем альтернативный блок `else`).

```
#include <iostream>
using namespace std;

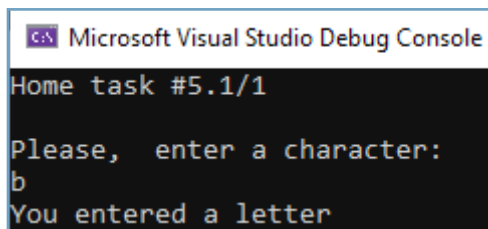
int main()
{
    cout << "Home task #5.1.1\n\n";
    int ch_code;
    char ch;
    cout << "Please, enter a character:\n";
    cin >> ch;

    ch_code = (int)ch;
```

```
if ((ch_code >= 65) && (ch_code <= 90)) ||  
    ((ch_code >= 97) && (ch_code <= 122))  
{  
    cout << "You entered a letter";  
}  
else if ((ch_code >= 48) && (ch_code <= 57))  
{  
    cout << "You entered a digit";  
}  
else if ((ch_code == 33) || ((ch_code >= 44) &&  
    (ch_code <= 46)) || (ch_code == 58) ||  
    (ch_code == 59) || (ch_code == 63))  
{  
    cout << "You entered a punctuation mark";  
}  
else  
{  
    cout << "You entered an unknown symbol";  
}  
return 0;  
}
```

Результаты работы программы (в консоли):

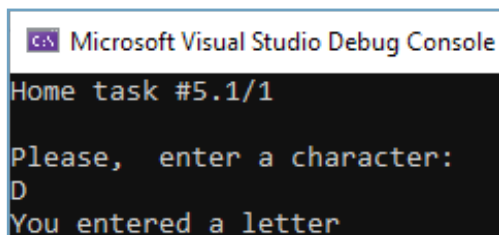
- Тест 1 — пользователь ввел строчную букву



```
Microsoft Visual Studio Debug Console  
Home task #5.1/1  
Please, enter a character:  
b  
You entered a letter
```

Рисунок 2

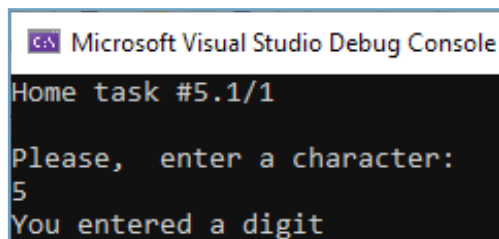
- Тест 2 — пользователь ввел заглавную букву



```
Microsoft Visual Studio Debug Console
Home task #5.1/1
Please, enter a character:
D
You entered a letter
```

Рисунок 3

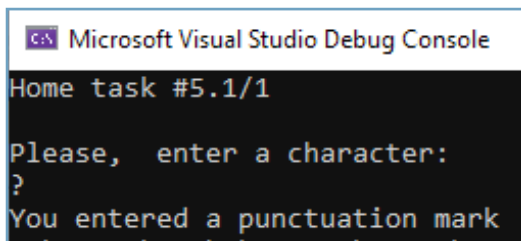
- Тест 3 — пользователь ввел цифру



```
Microsoft Visual Studio Debug Console
Home task #5.1/1
Please, enter a character:
5
You entered a digit
```

Рисунок 4

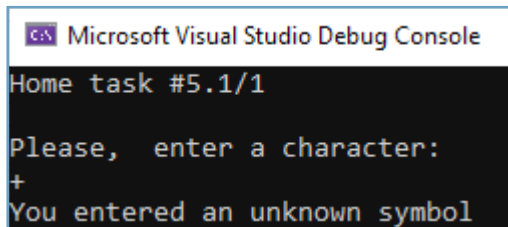
- Тест 4 — пользователь ввел знак пунктуации



```
Microsoft Visual Studio Debug Console
Home task #5.1/1
Please, enter a character:
?
You entered a punctuation mark
```

Рисунок 5

- Тест 5 — пользователь ввел непредусмотренный символ (категория «другое»)



```
C:\ Microsoft Visual Studio Debug Console  
Home task #5.1/1  
  
Please, enter a character:  
+  
You entered an unknown symbol
```

Рисунок 6

РЕШЕНИЕ ЗАДАНИЯ 2

Описание решения

После ввода пользователем данных о том, с какого на какой оператор он звонит (пользователь выбирает в меню код нужного оператора, вначале указывая свой оператор, а после вводит код второго оператора) и длительности разговора выполняем проверку, совпадают ли коды операторов.

Если совпадают, то выводим в консоль сообщение, что звонки внутри сети бесплатны.

Иначе организуем последовательность проверок для определения оператора исходящего звонка и стоимость одной минуты звонка за пределы сети для данного оператора.

Вычисляем общую стоимость разговора, умножая длительность на стоимость одной минуты звонка за пределы сети для данного оператора.

Если пользователь указал несуществующий код оператора, выводим сообщение об ошибке.

1. Выводим строку в консоль с описанием названия программы (по желанию, данная строка кода может быть пропущена), создаем необходимое число переменных — три целочисленных для хранения кодов двух операторов и длительности разговора, три вещественные для хранения стоимости одной минуты звонка за пределы сети для каждого из трех (в нашем примере) операторов (Orange, Vodafone, AT&T).

```
#include <iostream>
using namespace std;

int main()
```

```
{  
    cout << "Home task #5.1.2\n\n";  
  
    int userOp, companionOp, callMin;  
    float minOutKS, minOutV, minOutLC;  
    minOutKS = 2;  
    minOutV = 3;  
    minOutLC=2.5;  
  
    return 0;  
}
```

2. Выводим в консоль строку с приглашением пользователю ввести код своего оператора, код оператора собеседника и длительность разговора. Считываем введенные данные в соответствующие переменные.

```
#include <iostream>  
using namespace std;  
  
int main()  
{  
    cout << "Home task #5.1.2\n\n";  
  
    int userOp, companionOp, callMin;  
    float minOutKS, minOutV, minOutLC;  
    minOutKS = 2;  
    minOutV = 3;  
    minOutLC=2.5;  
  
    cout << "1- Orange\n";  
    cout << "2- Vodafone\n";
```



```
cout << "3- AT&T\n\n";
cin >> userOp;

cout << "Select your companion mobile operator:\n";
cout << "1- Orange\n";
cout << "2- Vodafone\n";
cout << "3- AT&T\n\n";
cin >> companionOp;
cout << "Enter the call duration:\n";
cin >> callMin;

return 0;
}
```

3. Проверяем, относятся ли пользователь и его собеседник к одному оператору. Если это так, то выводим в консоль сообщение, что звонки внутри сети бесплатны.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #5.1.2\n\n";

    int userOp, companionOp, callMin;
    float minOutKS, minOutV, minOutLC;
    minOutKS = 2;
    minOutV = 3;
    minOutLC=2.5;

    cout << "1- Orange\n";
```

```
cout << "2- Vodafone\n";
cout << "3- AT&T\n\n";
cin >> userOp;

cout << "Select your companion mobile operator:\n";
cout << "1- Orange\n";
cout << "2- Vodafone\n";
cout << "3- AT&T\n\n";
cin >> companionOp;

cout << "Enter the call duration:\n";
cin >> callMin;
if (userOp == companionOp)
{
    cout << "No fee!";
}

return 0;
}
```

4. Если пользователь и его собеседник не относятся к одному оператору, то организуем последовательность проверок для определения оператора исходящего звонка (оператора пользователя) и стоимость одной минуты звонка за пределы сети для данного оператора. Вычисляем общую стоимость разговора, умножая длительность на стоимость одной минуты звонка за пределы сети для данного оператора.

Так как выполниться может только одно из условий: оператор пользователя может быть только один из трех, то для реализации процедуры последовательной проверки

будем использовать оператор условного ветвления `if - else` `if - else`.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #5.1.2\n\n";

    int userOp, companionOp, callMin;
    float minOutKS, minOutV, minOutLC;
    minOutKS = 2;
    minOutV = 3;
    minOutLC=2.5;

    cout << "1- Orange\n";
    cout << "2- Vodafone\n";
    cout << "3- AT&T\n\n";
    cin >> userOp;

    cout << "Select your companion mobile operator:\n";
    cout << "1- Orange\n";
    cout << "2- Vodafone\n";
    cout << "3- AT&T\n\n";
    cin >> companionOp;

    cout << "Enter the call duration:\n";
    cin >> callMin;

    if (userOp == companionOp)
    {
        cout << "No fee!";
    }
}
```

```
else if (userOp == 1)
{
    cout << "Call cost - " << callMin* minOutKS;
}
else if (userOp == 2)
{
    cout << "Call cost - " << callMin * minOutV;
}
else if (userOp == 3)
{
    cout << "Call cost - " << callMin * minOutLC;
}

return 0;
}
```

5. Если пользователь указал несуществующий код оператора, выводим сообщение об ошибке (реализуем альтернативный блок `else`).

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #5.1.2\n\n";

    int userOp, companionOp, callMin;
    float minOutKS, minOutV, minOutLC;

    minOutKS = 2;
    minOutV = 3;
```

```
minOutLC=2.5;

cout << "1- Orange\n";
cout << "2- Vodafone\n";
cout << "3- AT&T\n\n";
cin >> userOp;

cout << "Select your companion mobile operator:\n";
cout << "1- Orange\n";
cout << "2- Vodafone\n";
cout << "3- AT&T\n\n";
cin >> companionOp;

cout << "Enter the call duration:\n";
cin >> callMin;

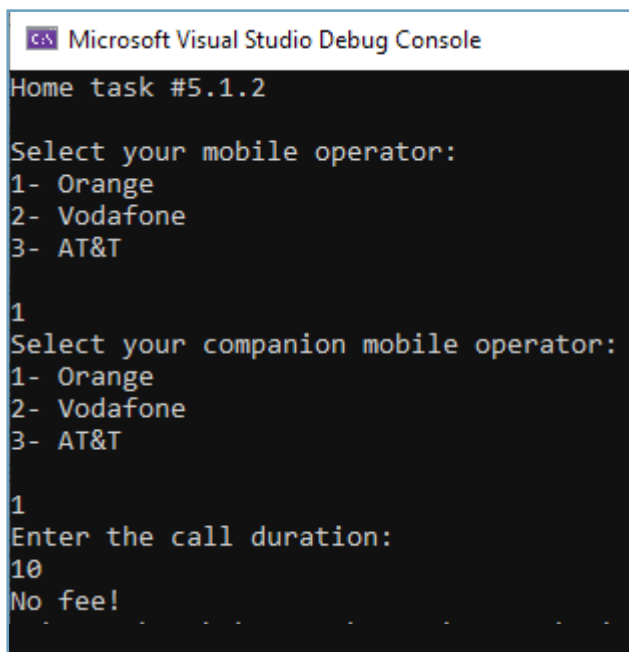
if (userOp == companionOp)
{
    cout << "No fee!";
}
else if (userOp == 1)
{
    cout << "Call cost - " << callMin* minOutKS;
}
else if (userOp == 2)
{
    cout << "Call cost - " << callMin * minOutV;
}
else if (userOp == 3)
{
    cout << "Call cost - " << callMin * minOutLC;
}
else
{

```

```
        cout << "Wrong input!";  
    }  
    return 0;  
}
```

Результаты работы программы (в консоли):

- Тест 1 — мобильные операторы пользователя и его собеседника совпадают

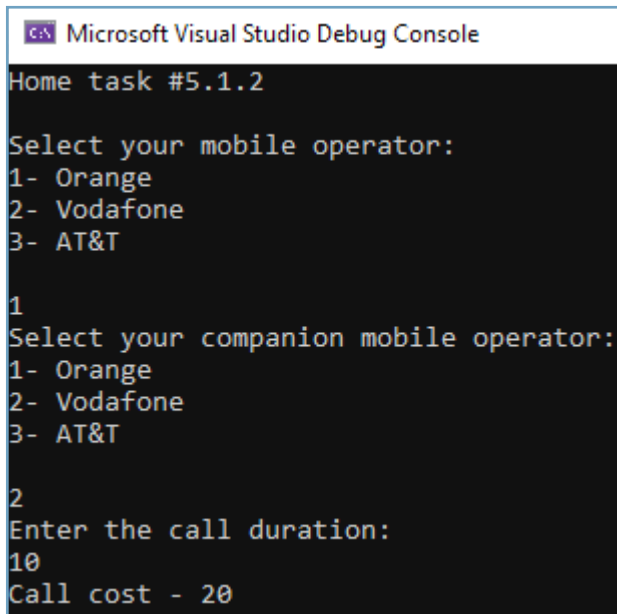


```
Microsoft Visual Studio Debug Console  
Home task #5.1.2  
  
Select your mobile operator:  
1- Orange  
2- Vodafone  
3- AT&T  
  
1  
Select your companion mobile operator:  
1- Orange  
2- Vodafone  
3- AT&T  
  
1  
Enter the call duration:  
10  
No fee!
```

Рисунок 7

- Тест 2 — мобильные операторы пользователя и его собеседника разные, стоимость разговора основывается на

стоимости одной минуты за пределы сети для оператора пользователя



```
Microsoft Visual Studio Debug Console  
Home task #5.1.2  
  
Select your mobile operator:  
1- Orange  
2- Vodafone  
3- AT&T  
  
1  
Select your companion mobile operator:  
1- Orange  
2- Vodafone  
3- AT&T  
  
2  
Enter the call duration:  
10  
Call cost - 20
```

Рисунок 8

РЕШЕНИЕ ЗАДАНИЯ 3

Описание решения

Для реализации меню пользователю необходимо вывести номера пунктов меню и их описание, чтобы он ввел нужный ему код пункта меню.

Если пользователь выбрал первый пункт меню, то запрашиваем у него желаемый доход Васи и количество опозданий.

Далее рассчитываем стоимость одной строки кода ($50\$/100=0.5\%$) и на ее основе вычисляем, сколько строк кода ему надо написать (без учета возможных штрафов):

$$\text{Количество строк кода} = \frac{\text{желаемый доход}}{\text{цена одной строки кода}} \quad (1)$$

Прежде чем рассчитывать сумму штрафов, нужно определить, а подвергается ли Вася штрафу за его количество опозданий. Если количество опозданий Васи меньше, чем 3, то штрафа нет. Иначе вычисляем, сколько раз будет применен штраф:

$$\text{Количество штрафных ситуаций} = \frac{\text{количество опозданий}}{3} \quad (2)$$

Одна штрафная ситуация — 20\$, рассчитываем общую сумму штрафов за допущенные Васей опоздания:

$$\text{Общая сумма штрафов} = \text{Количество штрафных ситуаций} * 20\$ \quad (3)$$

Желаемый доход уменьшается на сумму штрафа, значит, к количеству строк кода, обеспечивающих желаемый доход (без учета штрафов) нужно добавить количество строк кода, которые покроют штрафную сумму:

Количество строк кода +=

Общая сумма штрафов / цена одной строки кода (4)

Если пользователь выбрал второй пункт меню, то запрашиваем у него желаемый доход Васи и количество строк кода, написанное Васей.

Далее вычисляем уже заработанные Васей деньги, умножив цену одной строки кода на количество строк кода, написанное Васей:

уже заработанные деньги =

цена одной строки кода * количество строк кода (5)

Если желаемый доход Васи больше, чем Вася заработал, то опаздывать ему вообще нельзя.

Иначе находим объем возможной штрафной суммы (разницу между желаемым и реальным доходом) и на ее основе находим допустимое количество опозданий (известно, что штрафуются каждое третье опоздание и сумма одной штрафной ситуации 20\$):

допустимое количество опозданий= (6)

(желаемый доход – уже заработанные деньги) / 20 * 3

Но так как штрафуются только каждое третье опоздание, то Вася может опоздать на:

- допустимое количество опозданий;
- допустимое количество опозданий +1;
- допустимое количество опозданий +2.

Если пользователь выбрал третий пункт меню, то запрашиваем у него количество строк кода и количество опозданий.

Вычисляем уже заработанные Васей деньги по формуле (5).

Проверяем, подвергается ли Вася штрафу за его количество опозданий (не меньше ли 3). Если да, то рассчитываем общую сумму штрафов за допущенные Васей опоздания по формуле (3).

Если общая сумма штрафов больше или равна уже заработанной Васей сумме, то выдаем сообщение, что Вася не получит ничего.

Иначе вычисляем выплачиваемую Васе зарплату, вычитая из уже заработанной Васей суммы общую сумму штрафов:

$$\begin{aligned} &\text{Выплачиваемая зарплата} = \\ &\text{уже заработанная сумма} - \text{общая сумма штрафов} \end{aligned} \quad (7)$$

1. Выводим строку в консоль с описанием названия программы (по желанию, данная строка кода может быть пропущена), создаем необходимое число переменных: пять целочисленных (желаемый доход, количество строк кода, количество опозданий, сумма штрафов, код пункта меню), устанавливаем начальное значение суммы штрафов 0 и стоимость одной строки кода ($50\$/100=0.5\%$)

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #5.1.3\n\n";
    int income, lateNum, codeRows, penalty, userChoice;
    float rowPrice, temp;
    penalty = 0;
    rowPrice = 0.5;
    return 0;
}
```

2. Выводим в консоль описание пунктов меню и запрашиваем у пользователя номер желаемого пункта, сохраняем его в переменную

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #5.1.3\n\n";
    int income, lateNum, codeRows, penalty, userChoice;
    float rowPrice, temp;
    penalty = 0;
    rowPrice = 0.5;

    cout << "Select menu item:\n";
    cout << "1- income and number of lateness-> "
           "number of code lines\n";
    cout << "2- income and number of code lines -> "
           "number of lateness\n";
    cout << "3- >number of lateness and number
           "of code lines ->income\n";
    cin >> userChoice;

    return 0;
}
```

3. Реализуем проверку выбранного пользователем номера пункта меню, используя оператор выбора **switch**, который позволяет организовать проверку одной переменной на соответствие нескольких условий более эффективно и проще, чем набор из нескольких последовательных блоков **if - else if**.

Также после реализации задач выбранного пункта меню нужно остановить процесс проверки по остальным пунктам. Это реализуется путем использования оператора **break** в конце каждого кейса. Кейс **default** обрабатывает ситуацию ввода пользователем недопустимого номера пункта меню.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #5.1.3\n\n";
    int income, lateNum, codeRows, penalty, userChoice;
    float rowPrice, temp;
    penalty = 0;
    rowPrice = 0.5;

    cout << "Select menu item:\n";
    cout << "1- income and number of lateness-> "
           "number of code lines\n";
    cout << "2- income and number of code lines -> "
           "number of lateness\n";
    cout << "3- >number of lateness and number "
           "of code lines ->income\n";
    cin >> userChoice;

    switch (userChoice)
    {
        case 1:
        {
            break;
        }
    }
```

```
case 2:
{
    break;
}
case 3:
{
    break;
}
default:
    cout << "Wrong input!";
}
return 0;
}
```

4. Если пользователь выбрал первый пункт меню, то запрашиваем у него желаемый доход Васи и количество опозданий. Далее на основе стоимости одной строки кода вычисляем, сколько строк кода ему надо написать (без учета возможных штрафов) по формуле (1)

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #5.1.3\n\n";
    int income, lateNum, codeRows, penalty, userChoice;
    float rowPrice, temp;

    penalty = 0;
    rowPrice = 0.5;
```

```
cout << "Select menu item:\n";
cout << "1- income and number of lateness-> "
      "number of code lines\n";
cout << "2- income and number of code lines -> "
      "number of lateness\n";
cout << "3- >number of lateness and number
      of code lines ->income\n";
cin >> userChoice;
switch (userChoice)
{
case 1:
{
    cout << "Input income wished by user\n";
    cin >> income;
    cout << "How many times user was late?\n";
    cin >> lateNum;

    codeRows = income / rowPrice;
    break;
}
case 2:
{
    break;
}
case 3:
{
    break;
}
default:
    cout << "Wrong input!";
}
return 0;
}
```

5. Далее определяем, а подвергается ли Вася штрафу за его количество опозданий. Если количество опозданий Васи меньше, чем 3, то штрафа нет. Иначе вычисляем, сколько раз будет применен штраф и на основании этого количества рассчитываем общую сумму штрафов за допущенные Васей опоздания (формулы 2 и 3 соединены в одно выражение)

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #5.1.3\n\n";
    int income, lateNum, codeRows, penalty, userChoice;
    float rowPrice, temp;
    penalty = 0;
    rowPrice = 0.5;

    cout << "Select menu item:\n";
    cout << "1- income and number of lateness-> "
           "number of code lines\n";
    cout << "2- income and number of code lines -> "
           "number of lateness\n";
    cout << "3- >number of lateness and number "
           "of code lines ->income\n";
    cin >> userChoice;
    switch (userChoice)
    {
        case 1:
        {
            cout << "Input income wished by user\n";
            cin >> income;
            cout << "How many times user was late?\n";
```

```
    cin >> lateNum;
    codeRows = income / rowPrice;
    if (lateNum >= 3)
    {
        penalty = lateNum / 3 * 20;
    }
    break;
}
case 2:
{
    break;
}
case 3:
{
    break;
}
default:
    cout << "Wrong input!";
}
return 0;
}
```

6. Желаемый доход уменьшается на сумму штрафа, значит, к количеству строк кода, обеспечивающих желаемый доход (без учета штрафов) нужно добавить количество строк кода, которые покроют штрафную сумму

```
#include <iostream>
using namespace std;

int main()
{
```



```

cout << "Home task #5.1.3\n\n";
int income, lateNum, codeRows, penalty, userChoice;
float rowPrice, temp;
penalty = 0;
rowPrice = 0.5;

cout << "Select menu item:\n";
cout << "1- income and number of lateness-> "
        "number of code lines\n";
cout << "2- income and number of code lines -> "
        "number of lateness\n";
cout << "3- >number of lateness and number "
        "of code lines ->income\n";
cin  >> userChoice;

switch (userChoice)
{
case 1:
{
    cout << "Input income wished by user\n";
    cin  >> income;
    cout << "How many times user was late?\n";
    cin  >> lateNum;
    codeRows = income / rowPrice;
    if (lateNum >= 3)
    {
        penalty = lateNum / 3 * 20;
        codeRows = codeRows + penalty / rowPrice;
    }
    cout << "number of code lines: "
          << codeRows << "\n";
    break;
}
}

```

```
case 2:
{
    break;
}
case 3:
{
    break;
}
default:
    cout << "Wrong input!";
}
return 0;
}
```

7. Если пользователь выбрал второй пункт меню (сработал **case 2**), то запрашиваем у него желаемый доход Васи и количество строк кода, написанное Васей. Далее вычисляем уже заработанные Васей деньги, умножив цену одной строки кода на количество строк кода, написанное Васей:

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #5.1.3\n\n";
    int income, lateNum, codeRows, penalty, userChoice;
    float rowPrice, temp;
    penalty = 0;
    rowPrice = 0.5;
```

```
cout << "Select menu item:\n";
cout << "1- income and number of lateness-> "
      "number of code lines\n";
cout << "2- income and number of code lines -> "
      "number of lateness\n";
cout << "3- >number of lateness and number "
      "of code lines ->income\n";
cin >> userChoice;

switch (userChoice)
{
case 1:
{
    cout << "Input income wished by user\n";
    cin >> income;
    cout << "How many times user was late?\n";
    cin >> lateNum;

    codeRows = income / rowPrice;
    if (lateNum >= 3)
    {
        penalty = lateNum / 3 * 20;
        codeRows = codeRows + penalty / rowPrice;
    }
    cout << "number of code lines: "<< codeRows;
    cout << "\n";
    break;
}
case 2:
{
    cout << "Input income wished by user\n";
    cin >> income;
    cout << "Input number of code lines\n";
```

```
        cin >> codeRows;
        temp = codeRows * rowPrice;
        break;
    }
    case 3:
    {
        break;
    }
    default:
        cout << "Wrong input!";
    }
    return 0;
}
```

8. Сравниваемый желаемый доход и сумму, которую Вася заработал, то опаздывать ему вообще нельзя. Если желаемый доход больше, то выводим в консоль сообщение, что опаздывать вообще нельзя.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #5.1.3\n\n";
    int income, lateNum, codeRows, penalty, userChoice;
    float rowPrice, temp;
    penalty = 0;
    rowPrice = 0.5;

    cout << "Select menu item:\n";
    cout << "1- income and number of lateness-> "
           "number of code lines\n";
```

```
cout << "2- income and number of code lines ->
      number of lateness\n";
cout << "3- >number of lateness and number
      of code lines ->income\n";
cin >> userChoice;

switch (userChoice)
{
case 1:
{
    cout << "Input income wished by user\n";
    cin >> income;
    cout << "How many times user was late?\n";
    cin >> lateNum;

    codeRows = income / rowPrice;
    if (lateNum >= 3)
    {
        penalty = lateNum / 3 * 20;
        codeRows = codeRows + penalty / rowPrice;
    }
    cout <<"number of code lines: "<< codeRows;
    cout << "\n";
    break;
}
case 2:
{
    cout << "Input income wished by user\n";
    cin >> income;
    cout << "Input number of code lines\n";
    cin >> codeRows;
    temp = codeRows * rowPrice;
    if (income >= temp)
    {
```

```
        cout << "You are not allowed to be late!";
    }
    break;
}
case 3:
{
    break;
}
default:
    cout << "Wrong input!";
}
return 0;
}
```

9. Иначе находим объем возможной штрафной суммы (разницу между желаемым и реальным доходом), делим полученное значение на 20\$ (штраф за одно опоздание) и после умножаем на 3 (т.к. штрафуются только каждое третье опоздание).

Так как штрафуются только каждое третье опоздание, то Вася может опоздать на: допустимое количество опозданий; допустимое количество опозданий +1; допустимое количество опозданий +2.

Выводим полученные три допустимых количества опозданий в консоль.

```
#include <iostream>
using namespace std;

int main()
{
```

```
cout << "Home task #5.1.3\n\n";
int income, lateNum, codeRows, penalty, userChoice;
float rowPrice, temp;
penalty = 0;
rowPrice = 0.5;

cout << "Select menu item:\n";
cout << "1- income and number of lateness-> "
      "number of code lines\n";
cout << "2- income and number of code lines -> "
      "number of lateness\n";
cout << "3- >number of lateness and number "
      "of code lines ->income\n";
cin >> userChoice;

switch (userChoice)
{
case 1:
{
    cout << "Input income wished by user\n";
    cin >> income;
    cout << "How many times user was late?\n";
    cin >> lateNum;
    codeRows = income / rowPrice;
    if (lateNum >= 3)
    {
        penalty = lateNum / 3 * 20;
        codeRows = codeRows + penalty / rowPrice;
    }
    cout << "number of code lines: "<< codeRows;
    cout << "\n";

    break;
}
```

```
case 2:
{
    cout << "Input income wished by user\n";
    cin >> income;
    cout << "Input number of code lines\n";
    cin >> codeRows;
    temp = codeRows * rowPrice;
    if (income >= temp)
    {
        cout << "You are not allowed to be late!";
    }
    else
    {
        lateNum = (temp - income) / 20 * 3;
        cout << "You are allowed to be late ";
        cout << lateNum << " or " << lateNum + 1;
        cout << " or " << lateNum + 2 << " times";
    }
    break;
}
case 3:
{
    break;
}
default:
    cout << "Wrong input!";
}
return 0;
}
```

10. Если пользователь выбрал третий пункт меню, то запрашиваем у него количество строк кода и количество опозданий, сохраняем введенные данные в переменные.

Вычисляем уже заработанные Васей деньги по формуле (5).

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #5.1.3\n\n";
    int income, lateNum, codeRows, penalty, userChoice;
    float rowPrice, temp;
    penalty = 0;
    rowPrice = 0.5;
    cout << "Select menu item:\n";
    cout << "1- income and number of lateness-> "
           "number of code lines\n";
    cout << "2- income and number of code lines -> "
           "number of lateness\n";
    cout << "3- >number of lateness and number "
           "of code lines ->income\n";
    cin >> userChoice;

    switch (userChoice)
    {
    case 1:
    {
        cout << "Input income wished by user\n";
        cin >> income;
        cout << "How many times user was late?\n";
        cin >> lateNum;
        codeRows = income / rowPrice;
        if (lateNum >= 3)
        {
            penalty = lateNum / 3 * 20;
            codeRows = codeRows + penalty / rowPrice;
        }
    }
    }
```

```
    }  
    cout << "number of code lines: " << codeRows;  
    cout << "\n";  
    break;  
}  
case 2:  
{  
    cout << "Input income wished by user\n";  
    cin >> income;  
    cout << "Input number of code lines\n";  
    cin >> codeRows;  
    temp = codeRows * rowPrice;  
    if (income >= temp)  
    {  
        cout << "You are not allowed to be late!";  
    }  
    else  
    {  
        lateNum = (temp - income) / 20 * 3;  
        cout << "You are allowed to be late ";  
        cout << lateNum << " or " << lateNum + 1;  
        cout << " or " << lateNum + 2 << " times";  
    }  
    break;  
}  
case 3:  
{  
    cout << "Input number of code lines\n";  
    cin >> codeRows;  
    cout << "How many times user was late?\n";  
    cin >> lateNum;  
    temp = codeRows * rowPrice;  
    break;  
}
```

```
default:
    cout << "Wrong input!";
}
return 0;
}
```

11. Проверяем, подвергается ли Вася штрафу за его количество опозданий. Если да (т.е. количество допущенных Васей опозданий больше или равно 3), то рассчитываем общую сумму штрафов по формуле (3).

Иначе выводим в консоль уже заработанные Васей деньги, вычисленные на шаге 10.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #5.1.3\n\n";
    int income, lateNum, codeRows, penalty, userChoice;
    float rowPrice, temp;
    penalty = 0;
    rowPrice = 0.5;

    cout << "Select menu item:\n";
    cout << "1- income and number of lateness-> "
           "number of code lines\n";
    cout << "2- income and number of code lines -> "
           "number of lateness\n";
    cout << "3- >number of lateness and number "
           "of code lines ->income\n";
```

```
cin >> userChoice;

switch (userChoice)
{
case 1:
{
    cout << "Input income wished by user\n";
    cin >> income;
    cout << "How many times user was late?\n";
    cin >> lateNum;

    codeRows = income / rowPrice;
    if (lateNum >= 3)
    {
        penalty = lateNum / 3 * 20;
        codeRows = codeRows + penalty / rowPrice;
    }
    cout <<"number of code lines: "<< codeRows;
    cout << "\n";
    break;
}
case 2:
{
    cout << "Input income wished by user\n";
    cin >> income;
    cout << "Input number of code lines\n";
    cin >> codeRows;
    temp = codeRows * rowPrice;
    if (income >= temp)
    {
        cout << "You are not allowed to be late!";
    }
    else
    {

```

```
        lateNum = (temp - income) / 20 * 3;
        cout << "You are allowed to be late ";
        cout << lateNum << " or " << lateNum + 1;
        cout << " or " << lateNum + 2 << " times";
    }
    break;
}
case 3:
{
    cout << "Input number of code lines\n";
    cin >> codeRows;
    cout << "How many times user was late?\n";
    cin >> lateNum;
    temp = codeRows * rowPrice;
    if (lateNum >= 3)
    {
        penalty = lateNum / 3 * 20;
    }
    else
    {
        income = temp;
        cout << "You'll get " << income << "$\n";
    }
    break;
}
default:
    cout << "Wrong input!";
}
return 0;
}
```

12. Для ситуации, когда количество допущенных Васей опозданий больше или равно 3 (реализованная на шаге 11

проверка), выполняем следующую проверку: если общая сумма штрафов больше или равна уже заработанной Васей сумме, то выводим в консоль сообщение, что Вася не получит ничего.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #5.1.3\n\n";
    int income, lateNum, codeRows, penalty, userChoice;
    float rowPrice, temp;
    penalty = 0;
    rowPrice = 0.5;

    cout << "Select menu item:\n";
    cout << "1- income and number of lateness-> "
           "number of code lines\n";
    cout << "2- income and number of code lines -> "
           "number of lateness\n";
    cout << "3- >number of lateness and number "
           "of code lines ->income\n";
    cin >> userChoice;

    switch (userChoice)
    {
    case 1:
    {
        cout << "Input income wished by user\n";
        cin >> income;
        cout << "How many times user was late?\n";
        cin >> lateNum;
```

```
codeRows = income / rowPrice;
if (lateNum >= 3)
{
    penalty = lateNum / 3 * 20;
    codeRows = codeRows + penalty / rowPrice;
}
cout << "number of code lines: " << codeRows;
cout << "\n";
break;
}
case 2:
{
    cout << "Input income wished by user\n";
    cin >> income;
    cout << "Input number of code lines\n";
    cin >> codeRows;
    temp = codeRows * rowPrice;
    if (income >= temp)
    {
        cout << "You are not allowed to be late!";
    }
    else
    {
        lateNum = (temp - income) / 20 * 3;
        cout << "You are allowed to be late ";
        cout << lateNum << " or " << lateNum + 1;
        cout << " or " << lateNum + 2 << " times";
    }
    break;
}
case 3:
{
    cout << "Input number of code lines\n";
```

```
cin >> codeRows;
cout << "How many times user was late?\n";
cin >> lateNum;
temp = codeRows * rowPrice;

if (lateNum >= 3)
{
    penalty = lateNum / 3 * 20;

    if (penalty >= temp)
    {
        cout << "You'll get nothing\n";
    }
}
else
{
    income = temp;
    cout << "You'll get " << income << "$\n";
}
break;
}
default:
    cout << "Wrong input!";

}
return 0;
}
```

13. Иначе (если общая сумма штрафов меньше уже заработанной Васей суммы) вычисляем выплачиваемую Васе зарплату, вычитая из уже заработанной Васей суммы общую

сумму штрафов (по формуле 7) и выводим полученный результат в консоль.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #5.1.3\n\n";
    int income, lateNum, codeRows, penalty, userChoice;
    float rowPrice, temp;
    penalty = 0;
    rowPrice = 0.5;

    cout << "Select menu item:\n";
    cout << "1- income and number of lateness-> "
           "number of code lines\n";
    cout << "2- income and number of code lines -> "
           "number of lateness\n";
    cout << "3- >number of lateness and number "
           "of code lines ->income\n";
    cin >> userChoice;
    switch (userChoice)
    {
    case 1:
    {
        cout << "Input income wished by user\n";
        cin >> income;
        cout << "How many times user was late?\n";
        cin >> lateNum;
        codeRows = income / rowPrice;
        if (lateNum >= 3)
        {
```

```
        penalty = lateNum / 3 * 20;
        codeRows = codeRows + penalty / rowPrice;
    }
    cout << "number of code lines: " << codeRows;
    cout << "\n";
    break;
}
case 2:
{
    cout << "Input income wished by user\n";
    cin >> income;
    cout << "Input number of code lines\n";
    cin >> codeRows;
    temp = codeRows * rowPrice;
    if (income >= temp)
    {
        cout << "You are not allowed to be late!";
    }
    else
    {
        lateNum = (temp - income) / 20 * 3;
        cout << "You are allowed to be late ";
        cout << lateNum << " or " << lateNum + 1;
        cout << " or " << lateNum + 2 << " times";
    }
    break;
}
case 3:
{
    cout << "Input number of code lines\n";
    cin >> codeRows;
    cout << "How many times user was late?\n";
    cin >> lateNum;
```

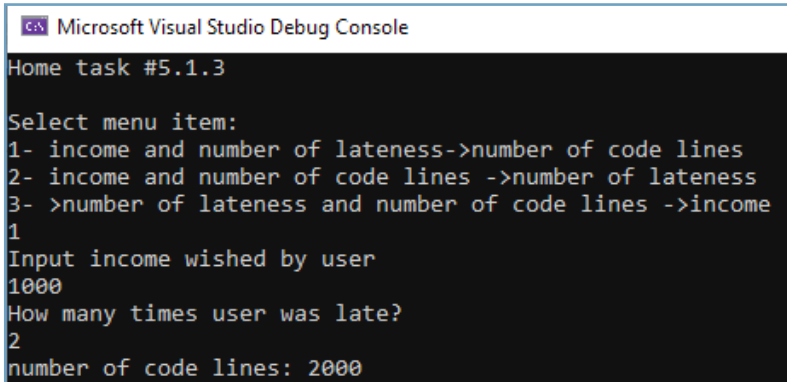
```
temp = codeRows * rowPrice;

if (lateNum >= 3)
{
    penalty = lateNum / 3 * 20;
    if (penalty >= temp)
    {
        cout << "You'll get nothing\n";
    }
    else
    {
        income = temp - penalty;
        cout << "You'll get " << income;
        cout << "$\n";
    }
}
else
{
    income = temp;
    cout << "You'll get " << income << "$\n";
}
break;
}
default:
    cout << "Wrong input!";
}

return 0;
}
```

Результаты работы программы (в консоли):

- Тест 1 — пользователь выбрал первый пункт меню, количество опозданий Васи меньше, чем 3 (2), штрафа нет.

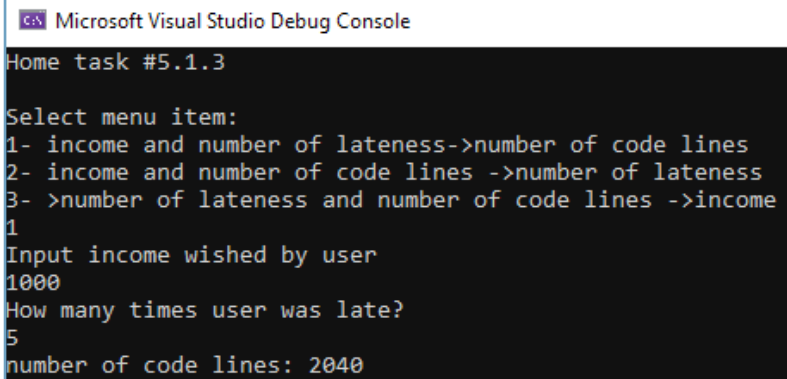


```
Microsoft Visual Studio Debug Console
Home task #5.1.3

Select menu item:
1- income and number of lateness->number of code lines
2- income and number of code lines ->number of lateness
3- >number of lateness and number of code lines ->income
1
Input income wished by user
1000
How many times user was late?
2
number of code lines: 2000
```

Рисунок 9

- Тест 2 — пользователь выбрал первый пункт меню, количество опозданий Васи больше, чем 3 (5), т.е. один штраф.

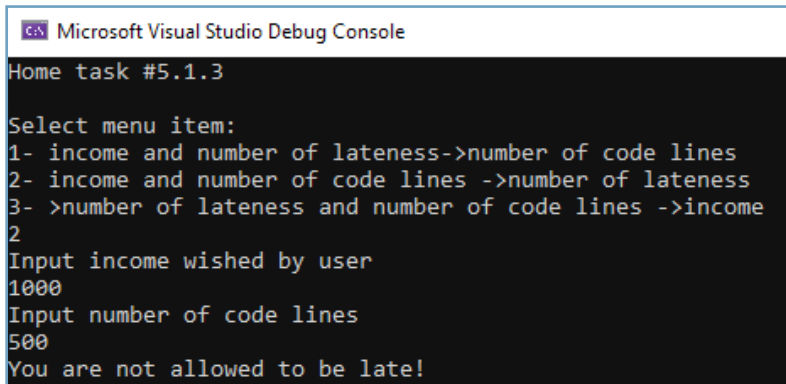


```
Microsoft Visual Studio Debug Console
Home task #5.1.3

Select menu item:
1- income and number of lateness->number of code lines
2- income and number of code lines ->number of lateness
3- >number of lateness and number of code lines ->income
1
Input income wished by user
1000
How many times user was late?
5
number of code lines: 2040
```

Рисунок 10

- Тест 3 — пользователь выбрал второй пункт меню, желаемый доход Васи больше, чем Вася заработал, опаздывать ему вообще нельзя.

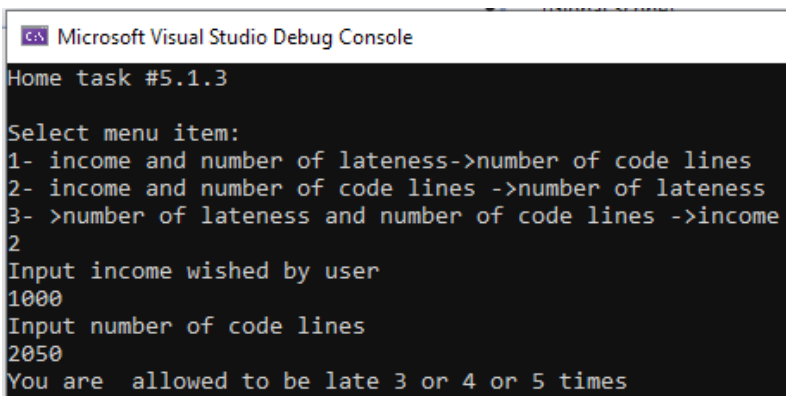


```
Microsoft Visual Studio Debug Console
Home task #5.1.3

Select menu item:
1- income and number of lateness->number of code lines
2- income and number of code lines ->number of lateness
3- >number of lateness and number of code lines ->income
2
Input income wished by user
1000
Input number of code lines
500
You are not allowed to be late!
```

Рисунок 11

- Тест 4 — пользователь выбрал второй пункт меню, желаемый доход Васи меньше, чем Вася заработал.

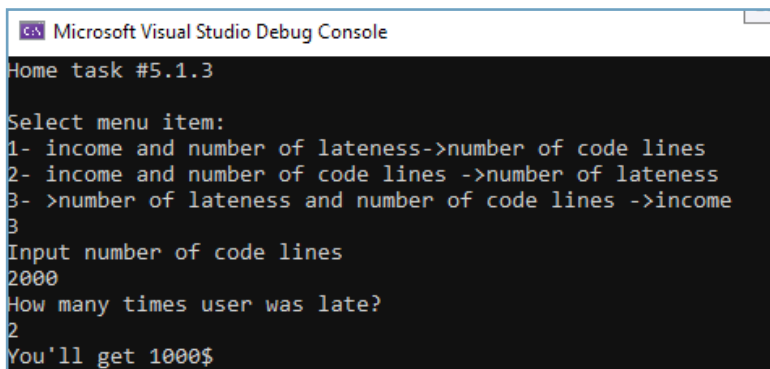


```
Microsoft Visual Studio Debug Console
Home task #5.1.3

Select menu item:
1- income and number of lateness->number of code lines
2- income and number of code lines ->number of lateness
3- >number of lateness and number of code lines ->income
3
Input income wished by user
1000
Input number of code lines
2050
You are allowed to be late 3 or 4 or 5 times
```

Рисунок 12

- Тест 5 — пользователь выбрал третий пункт меню, количество опозданий менее 3 (нет штрафов), ситуация, обратная тесту 1.



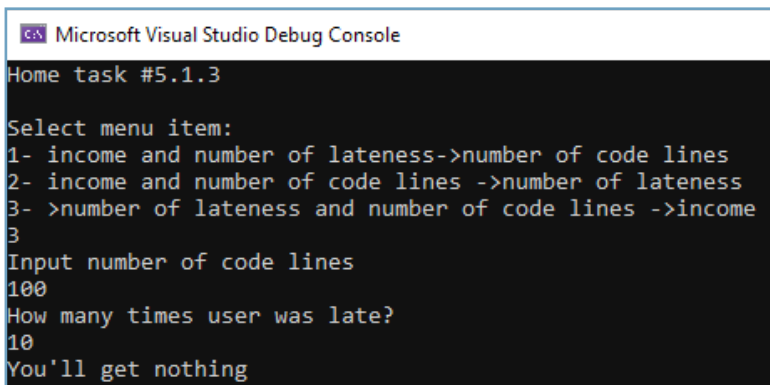
```
Microsoft Visual Studio Debug Console

Home task #5.1.3

Select menu item:
1- income and number of lateness->number of code lines
2- income and number of code lines ->number of lateness
3- >number of lateness and number of code lines ->income
3
Input number of code lines
2000
How many times user was late?
2
You'll get 1000$
```

Рисунок 13

- Тест 6 — пользователь выбрал третий пункт меню, количество опозданий более 3, общая сумма штрафов больше уже заработанной Васей суммы.



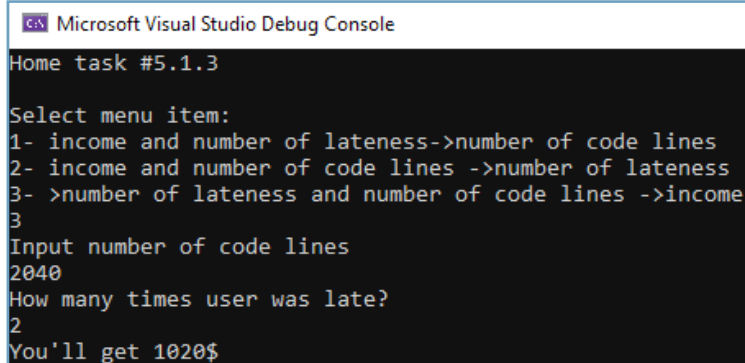
```
Microsoft Visual Studio Debug Console

Home task #5.1.3

Select menu item:
1- income and number of lateness->number of code lines
2- income and number of code lines ->number of lateness
3- >number of lateness and number of code lines ->income
3
Input number of code lines
100
How many times user was late?
10
You'll get nothing
```

Рисунок 14

- Тест 7 — пользователь выбрал третий пункт меню, количество опозданий более 3, общая сумма штрафов меньше заработанной Васей суммы.



```
Microsoft Visual Studio Debug Console  
Home task #5.1.3  
Select menu item:  
1- income and number of lateness->number of code lines  
2- income and number of code lines ->number of lateness  
3- >number of lateness and number of code lines ->income  
3  
Input number of code lines  
2040  
How many times user was late?  
2  
You'll get 1020$
```

Рисунок 15