РЕШЕНИЯ НА ЗАДАЧИТЕ

А. ОБИКОЛКА

1. Задава се дъжината **a** и ширината **b** на правоъгълника.
2. Пресмята се периметъра **P** на правоъгълника по формулата P=2*(a+b).
3. Съобщава се **P**.

Като използваме описаното решение и правилата за писане на програма на езика C++, можем да напишем следната програма:

```
//obikolka.cpp
# include<iostream.h>
void main( )
{
          float a,b,P;
          cin>>a>>b;
          P=2*(a+b);
          cout<<P<<endl;
}
```

В. СКЛАД

Най-лесно можем да решим задачата като в началото преобразуваме всички мерни единици в грамове, извършим изваждането на продаденото количество от наличното, след което преобразуваме оставащото количество в килограми и грамове. При това ще са необходими поне 6 променливи – по две за всяко количество: налично, продадено, оставащо.

Необходими величини:

Тук трябва добре да обмислим какъв ще е типът на променливите, с които ще работим. Както се вижда в условието, най-голямото количество стока, която може да се съхрани в склада е 10 тона, което ако се превърне в грамове е 10 000 000 грама. Това число очевидно не може да се събере нито в тип **int**, нито в **unsigned**. Все пак е необходимо да ползваме целочислен тип, тъй като за окончателното преобразуване на грамовете на оставащото количество в килограми ще ни трябва операцията за целочислено деление. Целочислен тип, който може да позволи използването на толкова големи числа е **long**. Ето защо дефинираме следните величини:

long nk, ng, pk, pg, ok, og;

където **nk** са наличните килограми, **ng** – наличните грамове, **pk** – килограмите на продаденото количество, **pg** – грамовете на продаденото количество, **ok** – килограмите на оставащото количество, **og**- грамовете на оставащото количество.

Въвеждаме входните данни:

cin>>nk>>ng>>pk>>pg;

Преобразуваме наличното и продаденото количество в грамове:

ng=nk*1000+ng;

pg=pk*1000+pg;

Пресмятаме оставащото количество в грамове:

og=ng-pg;

Превръщаме оставащото количество в грамове и килиграми:

ok=og/1000;

og=og%1000;

Извеждаме килограмите и грамовете на оставащото количество:

cout<<ok<<” kg “<<og<<”g\n”;

Програмата, решаваща тази задача има следния вид:

```
//sklad.cpp
#include<iostream.h>
void main()
        {
          long nk, ng, pk, pg, ok, og;
          cin>>nk>>ng>>pk>>pg;
          ng=nk*1000+ng;
          pg=pk*1000+pg;
          og=ng-pg;
          ok=og/1000;
          og=og%1000;
```

```
        cout<< ok<<” kg “<<og<<”g\n”;
}
```

C. САМОЛЕТ

Задачата се решава като към минутите на излитане се добавят минутите на продължителността на полета. След това получения резултат се преобразува в часове и минути, като часовете се прибавят към часа на излитане. Остава само да съобразим, че има възможност да се премине в следващото денонощие и тогава часа на кацане ще се получи като вземем остатъка при деление на получения час на 24.

1. Необходими величини:

Можем да решим задачата само с помощта на трите променливи обявени в условието, като ще се уговорим в променливите, указващи времето на излитане, да получим времето на кацане.

int h,m,ml;

2. Въвеждаме входните данни:

cin>>h>>m>>ml;

3. Прибавяме минутите на продължителността на полета към минутите на излитане:

m+=ml;

4. Преобразуваме получените минути в часове и минути, като часовете прибавяме към часа на излитане, а минутите записваме в променливата m:

h+=m/60;

m%=60;

5. Пресмятаме часа на кацане, като остатък при деление на 24 за да избегнем грешката при прехвърляне в другото денонощие:

h%=24;

6. Отпечатваме часа на кацане:

cout<<h<<”h “<<m<<”m\n”;

D. УРАВНЕНИЕ

1. Необходими величини:

За решението на задачата са необходими трите дробни променливи *a,b* и *x*:

float a, b, x;

2. В началото коефициентите на уравнението трябва да се въведат от клавиатурата.

cin>>a>>b;

3. Както вече беше обсъдено в темата “Алгоритми”, решението на уравнението зависи от това дали *a* е 0 или е различно от 0. Така, че на следващата стъпка се налага да се провери дали *a=0.*

a. Ако a=0, се проверява дали b=0.

3.1.1. Ако b=0, на екрана се отпечатва, че всяко x е решение на уравнението.

3.1.2. Ако b≠0, на екрана се отпечатва, че уравнението няма решение.

b. Ако a≠0, се пресмята x=b/a, след което на екрана се отпечатва неговата стойност.

Това на езика C++ се записва чрез следните оператори:

```
if(a==0)
          if(b==0)
             cout<<”всяко x е решение\n”;
            else
              cout<<”няма решение\n”;
        else
        {
          x=b/a;
          cout<<x<<endl;
        }
```

Тук веднага след условието на първия оператор **if** е вложен още един условен оператор. След клаузата му **else** е използван съставен оператор, тъй като алгоритъмът, описан по-горе предвижда две действия – пресмятане на x и отпечатването му на екрана.

Окончателния вид на програмата е следния:

```
#include<iostream.h>
void main()
 {
    float a, b, x;
```

```
    cin>>a>>b;
     if(a==0)
    { if(b==0)
      cout<<”всяко x е решение”<<endl;
         else
         cout<<”няма решение”<<endl;
    }
         else
         {
            x=b/a;
            cout<<”x=”<<x<<endl;
         }
 }
```

Е. ЛИЦА

1. Необходими величини:

int x; **float** a;

2. Първо се въвеждат двете числа **a** и **x**:

cin>>a>>x;

3. Проверката за това, коя от трите възможности е избрана, ще реализираме с оператор **switch** по следния начин:

```
 switch(x)
        {
         case 1: cout<<3*a<<endl; break;
         case 2: cout<<a*a<<endl; break;
         case 3: cout<<3.14*a*a<<endl; break;
          default: cout<<”Error!!!\n”; break;
        }
```

Окончателния текст на програмата е следния:

```
//area.cpp
#include<iostream.h>
void main()
         {
          int x; float a;
          cin>>a>>x;
          switch(x)
            {
             case 1: cout<<3*a<<endl; break;
             case 2: cout<<a*a<<endl; break;
             case 3: cout<<3.14*a*a<<endl; break;
               default: cout<<”Error!!!\n”; break;
            }
         }
```

F. ДЕТАЙЛИ

За да може да се произведе дадено изделие, за него трябва да има достатъчни количества и от двата детайла. За всеки от двата детайла ще изчислим **b1** и **b2**, съответно броя на изделията, които могат да се произведат с първия и втория детайл. Броя на произведените изделия се определя от по-малкото от двете числа **b1** и **b2**. В зависимост от това, кое изделие е избрано, програмата се разклонява в три посоки. Ето защо е удачно да използваме оператор **switch**.

1. Необходими величини:

За решението на задачата е необходима една целочислена променлива **i**, която ще приема стойности от 1 до 3 и ще определя номера на изделието, което ще се изработва. Освен това са необходими и целочислените величини **d1** и **d2**, които съдържат съответно наличното количество детайли от първия и втория вид. Накрая да отбележим, че са ни необходими и двете променливи **b1** и **b2**, които определят с кой детайл колко изделия могат да се произведат:

int i,d1,d2,b1,b2;

2. Първо трябва да се въведат предвидените от условието данни:

cin>>i>>d1>>d2;

3. След това, в зависимост от избраното изделие, трябва да се изчислят **b1** и **b2**:

```
switch(i)
         {
```

```
          case 1: b1=d1/2;b2=d1; break;
          case 2: b1=d1/4;b2=d2/2; break;
          case 3: b1=d1/5;b2=d2/4; break;
        }
```

4. Накрая ще се изведе по-малкото от числата **b1** и **b2**:

```
cout<<(b1<b2)?b1:b2<<endl;
```

Ето пълния текст на програмата, решаваща задачата:

```
//details.cpp
#include<iostream.h>
 void main()
        {
          int i,d1,d2,b1,b2;
          cin>>i>>d1>>d2;
          switch(i)
            {
              case 1: b1=d1/2;b2=d1; break;
              case 2: b1= d1/4;b2=d2/2; break;
              case 3: b1= d1/5;b2=d2/4; break;
            }
          cout<<(b1<b2)?b1:b2<<endl;
        }
```