1. Вычислите выражение $2110_3 - 48_9 + 5_{10}$. Ответ дайте в десятичной системе счисления.

Решение

Представим все числа в десятичной системе счисления.

$2110_3 = 2 \cdot 3^3 + 1 \cdot 3^2 + 1 \cdot 3^1 + 0 \cdot 3^0 = 66_{10}$

$48_9 = 4 \cdot 9^1 + 8 \cdot 9^0 = 44_{10}$

$66 - 44 + 5 = 27$ Ответ: 27

2. Михаил составляет 5-буквенные коды. В кодах разрешается использовать только буквы А, Б, В, Г, Д. При этом код не может начинаться с гласной и не может содержать двух одинаковых букв подряд. Сколько различных кодов может составить Михаил?

Решение

На первое место слова можно поставить любую из 5 букв, кроме буквы А. На каждое следующее место можно поставить любую из 5 букв, кроме той, которая стоит на предыдущем месте. Значит, всего Михаил может составить $4 \cdot 4 \cdot 4 \cdot 4 \cdot 4 = 1024$ кода.

Ответ: 1024

3. Автомат обрабатывает натуральное число N > 1 по следующему алгоритму.

1. Строится двоичная запись числа N.
2. Последняя цифра записи удаляется.
3. Если исходное число N было нечетным, в конец записи (справа) дописываются цифры 10, если четным – 01.
4. Результат переводится в десятичную систему счисления и выводится на экран.

Какое число нужно ввести в автомат, чтобы в результате получилось 2017?

Решение

Рассмотрим действие алгоритма в обратном порядке. Переведем число 2017 в двоичную систему счисления: $2017_{10} = 1111110000\ 1_2$. Последние две двоичные цифры – 01, следовательно, исходное число было четным. Рассмотрим числа, из которых можно получить натуральное число N.

$1111110000_{\ 2} = 1008_{10}$

$1111110001_{\ 2} = 1009_{10}$

Ответ: 1008

4. Для какого наименьшего целого неотрицательного числа A выражение $(2x + 3y > 30) \vee (x + y \le A)$ тождественно истинно, т.е. принимает значение 1 при любых целых неотрицательных x и y?

Решение.

Разделим исходное выражение $(2x + 3y > 30) \vee (x + y \le A)$ на две части:

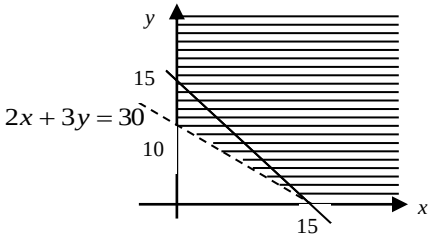
а) известная часть $2x + 3y > 30$;

б) неизвестная часть $x + y \le A$.

Определим условия истинности исходного выражения.

1. Если а) истинно, а б) ложно.
2. Если а) ложно, а б) истинно.
3. Если истинны одновременно а) и б).

Если для некоторых целых положительных x и y $(2x + 3y > 30)$, то исходное выражение истинно независимо от значения A (на рисунке выделено заштрихованной областью).



Поэтому будем рассматривать только область, в которой $2x + 3y \le 30$ (белая область, включая точки прямой $2x + 3y = 30$).

Необходимо найти такое A, чтобы для всех x и y из этой области выполнялось условие $x + y \le A$. Очевидно, A должно быть равно наибольшему возможному значению $x + y$ в этой области. На рисунке видно, что прямая $x + y = A$ должна быть правой границей области, определяемой условием $x + y \le A$ и должна проходить через точку (15, 0). Таким образом, наименьшее целое неотрицательное A равно 15.

Ответ: 15

5. Сколько различных решений имеет уравнение (K ∧ L) ∨ (M ∧ N) = 1, где K, L, M, N – логические переменные?

Решение.

Выражение истинно в трех случаях, когда (K ∧ L) и (M ∧ N) равны соответственно 01, 10, 11.

Составим таблицу решений.

1. "01":	K ∧ L = 0;		M ∧ N = 1.	
	K	L	M	N
1	1	0	1	1
2	0	1	1	1
3	0	0	1	1
2. "10":	K ∧ L = 1;		M ∧ N = 0.	
	K	L	M	N
1	1	1	1	0
2	1	1	0	1
3	1	1	0	0
3. "11":	K ∧ L = 1;.		M ∧ N = 1	
	K	L	M	N
1	1	1	1	1

Ответ: 7

6. Определите, что будет напечатано в результате работы следующего фрагмента программы:

Си++

```
#include <iostream>
using namespace std;
int main() {
   int s, k;
   s = 0, k = 0;
   while (s < 80)  {
      s = s + 2*k;
      k = k + 4;
    }
    cout << s << endl;
   return 0;
}
```

Решение.

Выполним указанный фрагмент программы, записывая в таблицу результаты

Шаг / Переменная	1	2	3	4	5
s	0	8	24	48	80
k	4	8	12	16	20

Ответ: 80

7. Чему будет равно значение, вычисленное рекурсивным алгоритмом при выполнении вызова F(5)?

си

```
int F(int n)
{
  if (n > 2)
    return F(n-1)+F(n-2)+F(n-3);
  else return n;
}
```

Решение

1. Вычисленное алгоритмом значение F равно.

F(5) = F(4) + F(3) + F(2) = F(3) + F(2) + F(1) + F(2) + F(1) + F(0) + 2= F(2) + F(1) + F(0) + F(2) + F(1) + F(2) + F(1) + F(0) + 2= 2 + 1 + 0 +2 + 1 +2 + 1 + 0 +2 =11

Ответ: 11

либо

2. Запишем все значения функции F:

F(0) = 0
F(1) = 1
F(2) = 2
F(3) = F(2) + F(1) + F(0) = 3
F(4) = F(3) + F(2) + F(1) = 6
F(5) = F(4) + F(3) + F(2) = 11

Ответ: 11

8. В программе используется одномерный целочисленный массив *A* с индексами от 0 до 9. Значения элементов равны 5; 1; 6; 7; 8; 8; 7; 7; 6; 9 соответственно, т.е. *A*[0] = 5; *A*[1] = 1 и т.д. Определите значение переменной ***c*** после выполнения следующего фрагмента программы.

Си++

```
c = 0;
for (i = 1; i <= 9; i++)
   if (A[i-1] >= A[i])
     {
        t = A[i];
        A[i] = A[i - 1];
        A[i - 1] = t;
     }
     else
        c++;
```

Решение.

Значение переменной ***c*** увеличивается на единицу, если значение элемента, стоящего на текущей позиции, меньше значения элемента, стоящего на предыдущей позиции. В противном случае соответствующие элементы меняются местами.

Элементы массива	**5**	**1**	**6**	**7**	**8**	**8**	**7**	**7**	**6**	**9**
		1 ↔ 5								
		5	6							
			6	7						
				7	8					
					8 ↔ 8					
						7 ↔ 8				
							7 ↔ 8			
								6 ↔ 8		
									8	9
Шаг (***i***)		***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***

Для заданного массива выполнено 5 перестановок, следовательно, значение переменной ***c*** = 9–5 = 4.

Ответ: 4

9. Напишите в ответе наименьшее значение входной переменной *k*, при котором программа выдаёт ответ 21.

Си++

```
#include <iostream>
using namespace std;
long f(long n) {
  return n * n * n;
}

long g(long n) {
  return n*n;
}

int main()
{
  long k, i;
  cin >> k;
  i = 1;
  while (f(i) <= k*g(i))
    i++;
  cout << i << endl;
  return 0;
}
```

Решение:

Цикл *While* завершится, когда выполнится неравенство $i^3 > k \cdot i^2$.

Задано условие окончания цикла: $21^3 > k \cdot 21^2$. Следовательно, последний шаг цикла определяется неравенством $20^3 \le k \cdot 20^2$. Находим *k* = 20.

Ответ: k = 20.

10. Дано целое число *N* (> 2) Реализовать алгоритм определения простоты числа *N*: число является простым, если у него всего два делителя: 1 и *N*.

Решение:

Для решения этой задачи реализуем следующий алгоритм:

*Опишем целочисленную переменную **N** (заданное число) и запросим ее значение с клавиатуры.*

*Опишем целочисленную переменную **i**, которую в дальнейшем будем использовать как переменную – параметр цикла и очередной делитель числа.*

*Опишем целочисленную переменную **k** – остаток от целочисленного деления заданного числа **N** на делитель **i**.*

*Инициализируем начальное значение переменной **i = 2**(первый делитель).*

*Инициализируем начальное значение переменной **k = 1** (число простое).*

*Организуем цикл вычисления очередного остатка **k**.*

*Цикл будет продолжаться пока **k >= 1** (**i** – не является делителем **N**) и **i <= N/2** (не перебраны все делители числа **N**).*

*После цикла значение переменной **k ≠ 0**, если заданное число **N** простое.*

Решение на языке Си представлено ниже:

```
#include <iostream>
using namespace std;
int main()
{
   long N, i;
   cin >> N;
   i = 2;  k = 1;
   while (k >= 1 && i <= N/2)
     {
        k = N % i;    i++;
     }
   if  (k==0)  cout << “Число составное” << endl; else cout << “Число простое” << endl;
   return 0;
}
```

11. Дан одномерный массив размера *N* (*N* — четное число, *N* <=100). Элементы массива – целые числа. Найти номера двух ближайших элементов из этого массива (то есть элементов с наименьшим модулем разности значений) и вывести эти номера в порядке возрастания. Вспомогательный массив не использовать.

Решение:

Для решения этой задачи реализуем следующий алгоритм. Опишем целочисленную переменную ***N*** *для хранения количества элементов массива. Опишем статический массив* ***X*** *из 100 элементов. Организуем ввод значения размерности массива с клавиатуры. Проверим, что введенное с клавиатуры значение не превышает размера статического массива. Для организации ввода элементов массива с клавиатуры используем цикл, который будет выполняться* ***N*** *раз.*

Для нахождения элементов массива, модуль разности которых минимален необходимо найти модули разности значений всех возможных пар элементов и выбрать из них минимальный. Для сохранения номеров указанных элементов опишем в программе переменные ***i1*** *и* ***i2****. Будем считать ближайшими элементами нулевой и первый элементы массива, то есть определим* ***i1 = 0, i2 = 1****, в переменной min сохраним модуль разности значений элементов* ***X[0]*** *и* ***X[1]****. Для перебора всех возможных пар элементов организуем два вложенных цикла. Внешний цикл будет перебирать элементы от нулевого до* ***N-1****(от первого до последнего) изменяя счетчик* ***i****, внутренний цикл будет перебирать элементы от* ***i****+1-го (чтобы исключить обращение к ранее просмотренным парам элементов) до последнего элемента, изменяя счетчик* ***j****. Во внутреннем цикле организуем вычисления модуля разности значений элементов* ***X[i]*** *и* ***X[j]*** *и сравнение найденного модуля с текущим значением переменной* ***min****. Если найденный модуль разности меньше текущего минимального значения, переопределим значения переменных* ***min****,* ***i1*** *и* ***i2****.*

После окончания циклов выведем значения переменных ***i1*** *и* ***i2****. Дополнительной проверки для сравнения найденных индексов можно не предусматривать, поскольку из-за описанной выше комбинации циклов переменная* ***i1*** *всегда будет меньше переменной* ***i2****. Реализация алгоритма на языке Си приведена ниже.*

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>

int main(int argc, char** argv) {
    int X[100],N,b,i,j,i1=0,i2=1;
    system("chcp 1251");
    printf("Введите размерность массива: ");
    scanf("%d",&N);
    if (N>100) {printf("Ошибка...");
                return 0;
    }
    printf("Вводите элементы массива: ");
    for(i=0;i<N;i++)
        scanf("%d",&X[i]);
    int min = abs(X[0]-X[1]);
    printf("\n");
    for(i=0;i<N;i++){
        for(j=i+1;j<N;j++){
            b = abs(X[i]-X[j]);
            if (b<min){ min = b; i1=i; i2=j;
            }
        }
    }
    printf("Ближайшие элементы - %d и %d \n",i1,i2);
    return 0;
}
```

12. Специальная камера, установленная на перекрёстке, фиксирует количество проезжающих автомобилей, и каждую минуту по каналу связи передаёт неотрицательное целое число — количество автомобилей, проехавших перекрёсток за эту минуту. Известно, что за минуту перекрёсток может проехать не более 100 автомобилей. Необходимо найти в заданной серии показаний максимальное количество автомобилей, проехавших перекрёсток в течение пяти подряд идущих минут. Максимальное количество показаний, которое может передать камера, не превышает 1440.

Напишите на любом языке программирования программу для решения поставленной задачи. Для получения максимального результата программа должна быть эффективна по времени и по используемой памяти.

Входные данные представлены следующим образом. В первой строке задаётся число *N* – общее количество переданных показаний. Гарантируется, что *N*>5. В каждой из следующих *N* строк задаётся одно положительное целое число – очередное показание камеры.

Пример входных данных:

8
5
12
27
10
4
50
7
16

Программа выводит только одно число – наибольшее количество автомобилей, проехавших перекресток за пять подряд идущих минут.

Пример выходных данных для приведённого выше примера входных данных:

103

Решение:

Для решения этой задачи реализуем следующий алгоритм.

*Опишем в программе переменную **N** для хранения количества входных данных. Для оптимизации программы по памяти будем хранить только серию из пяти текущих значений, для этого опишем в программе массив для хранения только пяти переменных – **X[5]**. Организуем цикл для ввода пяти первых значений в массив **X**. За максимальное количество машин, примем количество машин, проехавших перекресток за первые пять минут, для этого опишем переменную **max** и сохраним в ней сумму элементов массива. Это же значение сохраним в переменной **sum**. В дальнейшем переменную **sum** ,будем использовать для хранения количества машин, проехавших перекресток в течение теукщих пяти минут.*

*Для организации ввода оставшихся данных используем цикл, изменяющий счетчик **i**, который будет повторяться **N-5** раз. Опишем переменную **a**, в которую будем вводить значение количества машин, проехавших перекресток в **i-тую** минуту. В цикле будем выполнять следующие действия:*

- *ввод количества машин;*
- *уменьшение переменной **sum** на значение X[0] и увеличение на значение **a** (таким образом организовано вычисление количества машин, проехавших перекресток за текущие 5 минут);*
- *сдвиг элементов в массиве **X** на одну позицию влево и заполнение последнего элемента массива считанным значением **a**;*
- *сравнение значения **sum** с текущим значением максимальной суммы и изменение значения переменной max, если значение **sum > max**.*

*После выполнения суммы организуем вывод переменной **max**.*

Ниже представлен код программы на языке Си.

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char** argv) {
    int X[5],max,sum = 0,i,a, N, j;
    system("chcp 1251");
    printf("Введите количество показаний: ");
    scanf("%d", &N);
    printf("Вводите показания: ");
    for(i=0;i<5;i++){
          scanf("%d", &X[i]);
          sum=sum+X[i];
       }
    max = sum;
    for(i=0;i<N-5;i++){
      scanf("%d", &a);
      sum = sum-X[0]+a;
      if (sum>max) max = sum;
      // сдвиг элементов массива данных влево на одну позицию
      for (j=0;j<4;j++)
          X[j]=X[j+1];
      X[j] = a;
    }
    printf("Максимальное количество машин, проехавших перекресток за 5 минут - %d \n",max);
    return 0;
}
```