# **Примеры лекции 8**

**Обработка чисел**

## **while (a!=b) if (a>b) a-=b; else b-=a;**

# **Тема 7. Алгоритмы обработки чисел. 7.1.1 Обмен чисел**

# **int a,b, tmp; {1} tmp = a; a = b; b = tmp; {2} a = a + b; b = a - b; a = a - b; {3} a = a ^ b; b = a ^ b; a = a ^ b;**

# **7.1.2. Поиск количества цифр у числа int count( int n) { int c=0; while (n > 0) { n = n / 10; c++; } return c; }**

# **7.1.3. Переворачивание целого числа. int ReversNumber (int b)** { int ost ,rev=0; while(b>0) { ost=b%10; b=b/10; rev=rev\*10+ost; } return rev; }

# **7.1.4. Найти количество различных цифр у заданного натурального числа (не используя массивов).** **int SumDigit (int a)** { **int b,c rc=0; for (int i = 0; i<10; ++i) { c = 0; b = a; while(b>0) { if(b%10 == i) c++; b /=10;} if(c!=0) rc++; } return rc; }**

# **7.2.** **Алгоритм Евклида** Заданы *a* „y *b* - целые числа, не равные одновременно нулю, и последовательность чисел *r1* < *r2* < *r3* <….< *rn* , определена тем, что каждое *rk* это остаток от деления предпредыдущего числа на предыдущее, а предпоследнее делится на последнее нацело, т. е. a = bx1 + r1 b1 = r1x1 + r2 ... bn-1 = rn-2xn-1 + rn-1 bn = rn-1xn + 0 => rn-1 = НОД(a,b)

# ***Найти НОД (645; 381).*** Разделим с остатком 645 на 381: 645=381·1+264. разделим 381 на 264: 381=264·1+117. разделим 264 на 117: 264=117·2+30. разделим 117 на 30 : 117=30·3+27. разделим 30 на 27 : 30=27·1+3. разделим 27 на 3 : 27=3·9 +0, т. е. 27 делится на 3 без остатка. **Таким образом, НОД (645; 381) = 3. последнему отличному от нуля остатку.**

# **{ 7.2.1} Дана пара натуральных чисел (a, b). Шаг 1. Поделить a на b с остатком r Шаг 2. Если r = 0, то НОД (a, b) = b Шаг 3. Если r <>0, то перейти к шагу 1 с парой (b, r)**

# **int Nod (int a, int b) { int r; while (b != 0) { r = a % b; a = b; b = r; } return r;** }

# **Упр. 7.1 Если цикл записать** **так: int Nod (int a, int b)** { **do { if (a>b) a=a % b; else b=b % a; while ((a !=0) && (b!=0)); } return ???; }**

# **7.3.** **Бинарный алгоритм Евклида** вычисляет наибольший общий делитель двух целых чисел. **Вычисления проводятся на основании следующих свойств НОД: 1. НОД(2m,2n) = 2 НОД(m ,n), 2. НОД(2m,2n+1) = НОД(m,2n+1), 3. НОД(-m,n) = НОД(m,n)**

# **Алгоритм 4. НОД(0, n) = n; НОД(m, 0) = m; 5. Если m,n чётные, тогда НОД(m, n) = 2\*НОД(m/2, n/2) 6. Если m чётная, тогда НОД(m, n) = НОД(m/2, n) 7. Если n чётная, тогда НОД(m, n) = НОД(m, n/2).**

# **8. Если m,n нечётные и m > n, тогда НОД(m, n) = НОД(m - n, n) 9. Если m,n нечётные и m < n, тогда НОД(m, n) = НОД(n - m, m) 10. Если m = n, тогда НОД(m, n) = m;**

# **int BinNod(int x ,int y) { if (x == 0 || y == 0) //одно из чисел=0 nod = x | y; //НОД=ненулевому else {/\* shift = наибольшая степень 2, чтобы x и y разделились на 2shift \*/ for (shift = 0; ((x | y) & 1) == 0; ++shift) // цикл работает пока оба числа четные { x >>= 1; // деление на 2 y >>= 1; *//* деление на 2 }**

# **while ((x & 1) == 0) // пока x не станет нечетным делим на 2 x >>= 1; do { while ((y & 1) == 0) // пока y не станет нечетным тоже делим на 2 y >>= 1; /\* теперь x и y оба нечетны, разность их четная, положим x = min(x,y), y = разнос-ти большего и меньшего, деленной на 2\*/**

# **else { unsigned int diff = x - y; x = y; y = diff; } y >>= 1; } while (y != 0); nod = x << shift; return nod; } }**

# **7.4. Простые числа. int prime (int a) { int i, flag; double r; if ((a==2)||(a==3)) flag =1; if (a==1) flag =-1;**

# **else { i=2; flag =1; r=a; n= (floor(sqrt (r)); while ((i< =n) && ( flag ) ) if (!(a%i)) flag=0; else ++i; return flag; }**

# **7.5. Числа Мерсенна** В XVII веке французский математик М. Мерсенн определил последовательность чисел вида: Mn = 2n − 1 Эта последовательность получила наименование «**чисел Мерсенна**»: 1, 3, 7, 15, 31, 63, 127, 255, 511, 1023, ... Иногда числами Мерсенна называют числа Mp с простыми индексами p. Эта последовательность: 3, 7, 31, 127, 2047, 8191, 131071, 524287, 8388607,...

# М11=2047=23\*89 простым не является. До 1750 года было найденовсего 8 простых чисел Мерсенна: М2, М3, М5, М7, М13, М17, М19, М31. То, что М31 - простое число, доказал в 1750 году Л. Эйлер. В 1876 году французский математик Эдуард Люка установил, что число М127=170141183460469231731687303715884105727- простое. В 1883 г. Сельский священник Пермской губернии И.М.**Первушин** без всяких вычислительных приборов доказал, что число М61=**2305843009213693951** является простым. Позднее было установлено, что числа М89 и М107 -простые. Использование ЭВМ позволило в 1952-1964 годах доказать, что числаМ521, М607, М1279, М2203, М2281, М3217, М4253, М4423, М2689, М9941, М11213- простые. **УПР 7.2.** Какое наибольшее простое число найдено сегодня

# **{ //поиск чисел Мерсенна for (int i=1;i<=m;i++) { int r=1,x=2; int n=i; while (n != 0) { if (n % 2 == 1) r=r\*x; x=x\*x; n=n / 2; } r=r-1; cout<<r<<endl; } }**

# **Или**

# #include <iostream>

# using namespace std;

# int main ()

# {

# int n ;

# cin>>n;

# int k = 0;;

# for (int i = 1; i <= n; i++)

# {

# k = (1 << i ) - 1;

# cout << k <<" ";

# }

# system ("pause");

# return 0;

# }

**7.6 Дружественные числа** Это такие два числа, каждое из которых равно сумме делителей второго. Наименьшие из дружественных чисел 220 и 284 были известны еще пифагорейцам, которые считали их символом дружбы.   
 Следующая пара дружественных чисел 17296 и 18416 была открыта французским юристом и математи-ком Пьером Ферма в 1636 году, а следующие числа находили Декарт, Эйлер и Лежандр.  
 Шестнадцатилетний итальянец Никколо Паганини в 1867 году потряс математический мир сообщением о том, что числа 1184 и 1210 дружественные! Эту пару, ближайшую к 220 и 284, проглядели все знаменитые математики, изучавшие дружественные числа.

# **7.7. Совершенные числа.** **Большой интерес к простым числам Мерсенна вызван их тесной связью с совершен-ными числами. Натуральное число Р называется совершенным, если оно равно сумме всех своих делителей кроме Р. Евклид доказал, что если р и 2р-1 - простые числа, то число Р=2р-1(2р-1) является совершенным**. **УПР 7.3. Доказать!**

# **6 28 496 8128 33550336 8589869056 137438691328 2305843008139952128**

# Первые четыре совершенных числа приведены в *Арифметике* **Никомаха Геразского**. Числа Р2=6 и Р3=28 были известны ещё пифагорейцам. Числа Р5=496 и Р7=8128 нашел Евклид. Пятое число **Р13** =**33550336** обнаружил немецкий математик **Региомонтан** (XV век). В XVI веке немецкий ученый **Шейбель** нашел еще два числа: **Р17 =8589869056 и Р19= 137438691328**. **Р31=2305843008139952128**. **В начале XX в. были найдены еще 3 совершенных числа (для р = 89, 107 и 127). В дальнейшем поиск затормозился вплоть до середины XX в., когда с появлением компьютеров стали возможными огромные вычисления. На февраль 2008 г. – 44, на апрель 2010 -47 совершенных числа. Нечётных совершенных чисел до сих пор не обнаружено, однако не доказано и то, что их не существует. Неизвестно также, бесконечно ли множество всех совершенных чисел**

# До сих пор остаётся загадкой, как Мерсенн смог высказать правильное утверждение, что числа Р17, Р19, Р31 являются совершенными. Позднее было обнаружено, что почти за сто лет до Мерсенна числа Р17, Р19 нашел итальянский математик Катальди - профе-сор университетов Флоренции и Болоньи. Считалось, что божественное провидение предсказало своим избранникам правильные значения этих чисел**.** Однако и с математической точки зрения чётные совершенные числа по-своему уникальны. Все они - треугольные. Сумма величин, обратных всем делителям числа, включая само число, всегда равна двум. Остаток от деления совершенного числа, кроме 6, на 9 равен 1. В двоичной системе совершенное число Рр начинается р единицами, потом следуют р-1 нулей. Р2=110, Р3=11100, Р5 =111110000, Р7 =1111111000000 и т.д. Последняя цифра чётного совершенного числа или 6, или 8, причём, если 8, то ей предшествует 2.

# Леонард Эйлер доказал, что все чётные совершенные числа имеют вид (2р-1)\* Мр, где Мр-простое число Мерсенна. Однако до сих пор не найдено ни одного нечётного совершенного числа. Высказано предположение (БрайенТакхерман,США), что если такое число существует, то оно должно иметь не менее 36 знаков.

# **Замечательное свойство совершенных чисел, открытое, как полагают, учениками Пифагора гласит: если число р=1+2+4+...+2n - простое, то число 2n\*p совершенное.** **Упр 7.2 Пример поиска совершенных чисел смотри**

# **#include <iostream> using namespace std; int step(int x,int n) {…;} int prime (int a) {…;} void main() { int p=2, i=1, n, ch; cin>>n; while (i<=n) { if (prime(p)&&(prime (step(2,p)-1))) ) { ch=step(2,p-1)\*(step(2,p)-1); cout <<ch<<endl; i++; } p++;}}**

# **7. 7. Числа Армстронга n-значное число называется числом Армстронга, если оно равно сумме n-ых степеней своих цифр. 153, 370, 371, 407, ... 153=13+5 3 +3 3**

# 7.**8 Числа Смитта** Число называется числом Смита, если сумма цифр числа равна сумме цифр разложения этого числа на простые множители. Например: **4937775 - число Смита** 4937775=3 х 5 х 5 х 65837 Сумма цифр числа - 42  Сумма цифр произведения - 42 Числа Смита: **4, 22, 27, ...** На интервале (0, 10 000) - 376 чисел Смита На интервале (0, 100 000) - около 3300 чисел Смита На интервале (0, 1 000 000) - 29928 чисел Смита Число чисел Смита бесконечно

//Проверка на число Смита, считаем сумму цифр, затем считаем сумму цифр

//простых делителей и сравниваем, если совпадают, то это число Смита

# #include <iostream>

# using namespace std;

# int Summa(\_\_int64 a) {

# int x=0;

# while(a) {

# x+=a%10;

# a/=10;

# }

# return(x);

# }

# int main(){

# \_\_int64 i=2,b,a,s=0;

# cout<<"Vvedite chislo: ";

# cin>>a;

# b=Summa(a);

# while(i<=a) {

# if (a%i==0) {

# s+=Summa(i);

# a/=i;

# i--;

# }

# i++;

# }

# if (b==s)

# cout<<"Dannoe chislo - chislo Smita"<<endl;

# else

# cout<<"Dannoe chislo - NE chislo Smita"<<endl;

# return 0;

# }

# **7.9. Числа Фибоначчи Число называется числом Фибоначчи, если оно является одним из членов последовательности: *f*n = *f*n-1 + *f*n-2 где *f*0 = 1 и *f*1 = 1.**

# Можно также определить n-й член ряда Фибоначчи, непосредственно подсчитав выражение:

# Иногда термином **самовлюблённые** числа называют любой тип чисел, которые равны некоторому выражению от их собственных цифр. Числа Фридмана, Числа Брауна. Совершенные числа. Дружественные числа

# **7.10. Для заданного натурального числа определить количество единичных бит в его представлении.**

# Суть алгоритма в том, что заводится битовая "маска" (переменная mask), содержащая единственный ненулевой бит, позиция которого при работе цикла меняется от младшего к старшему (либо в обратную сторону). При единичном значении очередного бита проверяемого числа (переменная ***a*),** логическое умножение очередного значения маски на содержимое проверяемого числа маску не меняет. Этот факт и проверяется для увеличения счетчика ***num*.**

# **int Count\_Bit1 ( int a)** { int bit; int mask = 1; **//двоичное число 0000 0000 0000 0000 0000 0000 0000 0001** int num = 0; **for ( int i = 0; i <=31; i++)** { //проверка битовой 1 в соответствующем разряде} **bit=mask & a; if (bit == mask) num++; mask = mask << 1 ;** } **return num;}**

# **7.11. У заданного натурального числа удалить младшей цифру int Count\_Bit2 ( int a) {** int tmp = a, num = 0; while (tmp > 0) { **tmp= tmp – (tmp & (tmp ^ (tmp - 1)));** **num++;** } **return num; }**

# **7.12 По заданному натуральному числу ( 4 байта ) получить новое число, переставив младшую цифру исходного числа на место впереди старшей. (Например, из входного значения 789 должно получиться 978.)**

# Возможный алгоритм решения состоит из следующих шагов: ·         определяется количество цифр в исходном числе (при ***n=789 = 3***); ·         исходное число обрезается перед младшим разрядом **tmp,** а значение младшего разряда запоминается (***9 → d*);** ·         умножением на ***10k-1*** выделенная цифра сдвигается в нужный разряд числа (***9\*102=900***); ·         к результату добавляется обрезанное значение (***900+78=978 → nr*);** ).

# **int revers1( int n) { int tmp, d, k, nr; k= count( n); d = n % 10; tmp = n / 10; nr = d \* pow(10, k - 1) + tmp; }**

# **7.13. Определить местоположение младшего единичного бита в заданном натуральном числе, представленном в машинном виде.** Для решения достаточно остановиться при обнаружении первой (справа) единицы и зафиксировать значение управляющей переменной цикла с параметром. Вообще говоря, в этом случае, вместо цикла с параметром, естественнее выглядит цикл с постусловием

# do { bit=mask & a; num++; mask = mask << 1 ; } while (bit == 0); cout<<num; Oшибка если а=0

# Задачи для самостоятельного решения 1.      В битовом представлении натурального числа перенести младший бит, поставив его перед старшим единичным битом. 2.      В битовом представлении натурального числа перенести m младших бит, подставив их перед старшими (m не превосходит длины двоичного представления числа).

# **7.14. Перевод чисел в различные нумерации. Перевести натуральное число *n* в систему счисления с основанием *p*.** **Очевидно, в результате нужно получить разложение числа *n* в позиционной нумерации *Ap*, то есть символьную строку вида *akak-1*... *a1a0* (где *ai* принадлежит *Ap*), которая соответствует сумме *n=ak\*pk+ak-1\*pk-1+... +a1\*p1+a0*. В частном случае при *p=10* мы как раз и имеем "обычное" десятичное разложение.**

# **Программная реализация алгоритма связана с некоторыми ограничениями. Поскольку в вычислениях применяются операции / и %, используемые в языке программирования для обработки десятичных чисел, да и результат тоже представлен как десятичная запись - символами из *A10*, мы ограничимся диапазоном *2≤p≤9*.**

# **int a,r; { // ввод n a = 0; r = 1; while (n >= 1) { a = a + (n % p) \* r; r = r \* 10; n = n / p; } // Decimal\_P = a**

# **4** Наименьшее число цветов для раскраски карты на плоскости. Тетраэдальное число. 5 Число Платоновых многогранников. Пятое число из последовательности Фибоначчи. Пирамидальное число. **6** =3! Наименьшее совершенное число. Треугольное число. **7** Наименьшее число сторон многоугольника, которым нельзя замостить плоскость. Шестиугольное число. **8** Наибольший куб в последовательности Фибоначчи. Имеет горизонтальную и вертикальную оси симметрии. **9** Максимальное число кубов, необходимое для представления в виде их суммы любого положительного целого числа. **10** Основание нашей системы счисления. Число топологически различных фигур из 5 спичек. Тетраэдальное и треугольное число.

# **11** Наибольшее количество кусков, на которые делят круг 4 прямые линии. Имеет горизонтальную ось симметрии. **12** Наименьшее число, имеющее 4 делителя. Количество плиток пентамино. **13** Число Архимедовых многогранников. Число из последова-тельности Фибоначчи. Перестановочное (с 31) простое число. **14** Четвертое число Каталана. Пирамидальное число. **15** Четвертое число последовательности Белла. Треугольное число. Произведение первых трех нечетных чисел. Количество сочетаний четырех чисел из шести. **16** Единственное число (кроме 1), выражаемое в форме xy=yx , а именно 24=42. **17** Количество вариантов узоров, построенных с использованием сдвигов, поворотов и отражений. Перестановочное (с 71) простое число. **18** Единственное число, равное удвоенной сумме его цифр.