ЗАДАНИЕ ПО ИНФОРМАТИКЕ
ВАРИАНТ 73101 для 10 класса

Для заданий 1, 2, 4, 5 требуется разработать алгоритмы на языке блок-схем, псевдокоде или естественном языке

1. Для проверки, является ли большое целое простым, может использоваться вероятностный тест Ферма. Пусть $p > 2$ – проверяемое число. Тогда:
   - случайно выбираем a: $2 \le a \le p - 2$;
   - если $a^{p-1} \neq 1 \ \{\mathrm{mod}\ p\}$, то p – составное.

   В тесте Ферма эти проверки выполняются для t случайно выбираемых a.
   Написать алгоритм проверки вводимого числа на простоту по тесту Ферма.
   Примечание: $x = y \ \{\mathrm{mod}\ n\}$, если существует целое k, для которого $x = y + k \cdot n$.

Схема решения. В цикле t раз выбираем число a в заданном диапазоне. Для каждого a проверяем условие $a^{p-1} \neq 1 \ \{\mathrm{mod}\ p\}$. Для того чтобы при возведении в степень не получилось слишком большого числа (выходящего за пределы чисел, представимых в компьютере), можно применять операцию вычисления остатка от деления после каждого умножения на a. Если для какого-то значения a результат не равен 1, то число – составное, и проверку можно прекращать.

2. В пансионате для спортивного досуга детей оборудована специальная площадка с большим числом крупных клеток L, выложенных в дорожки одинаковой длины. По дорожкам (от начала до конца) с клетки на клетку любят прыгать отдыхающие дети. На каждой клетке нарисован вес – натуральное число. Выигрывает тот ребенок, который при прыжках набрал минимальный суммарный вес. В игре принимали участие $M \le L$ детей, прыгающих за один ход на 1, 3, 4 или 5 клеток. Предложите наиболее оптимальный способ обработки и хранения информации для моделирования ситуации (например, для определения победителя). Примечание: прыгать на 2 клетки нельзя.

Схема решения. Для хранения данных будем использовать матрицу из $L \times L$ элементов, в элементах которой записан вес каждой клетки. Результаты детей запишем в виде массива из $L \times M$ элементов, причём элемент равен 1, если ребёнок прыгал на данную клетку, и 0 – в противном случае. Также надо записать номер дорожки, по которой он прыгал. Чтобы получить итоговый результат, надо сложить веса клеток, на которые прыгал каждый ребёнок. Для проверки соблюдения правил будем вычислять разницу между индексами элементов в каждой строке второй матрицы, содержащих значение 1. Для этого положим сначала номер предыдущей использованной клетки 0, затем при нахождении в массиве значения 1 вычисляем разницу. Если она не равна 1, 3, 4 или 5, значит, правила была нарушены. Иначе прибавляем к общему результату вес текущей клетки и запоминаем номер текущей клетки как номер предыдущей.

3. Десятиклассник Сережа любит играть с калькулятором. Он часто сначала делит вещественные числа a и b друг на друга, а затем результат умножает на b. Выполнив эти действия много раз (сначала много делений, а затем столько же умножений), Сережа получил в результате некоторое число. Будет ли оно исходным? Объясните, почему?

Ответ. Не будет. Возникающая в процессе вычислений погрешность, обусловленная округлением вещественных чисел при представлении в ЭВМ, изменит результат.

4. Не используя дополнительный массив или простые методы сортировок, найти в матрице номера двух первых минимальных элементов.

Схема решения. Найдём минимальный элемент матрицы. Проверим, сколько таких элементов существует в матрице. Если их 2 и больше, значит, задача решена. Если нет, ещё раз найдём минимальный элемент из тех элементов, которые больше первого минимума.

5. Число Фибоначчи – натуральное число, удовлетворяющее следующим соотношениям: $F_0 = 1$, $F_1 = 1$, $F_n = F_{n-1} + F_{n-2}$, $n \geq 2$. Даны целые числа n и m ($1 \leq n \leq 10^{18}$, $2 \leq m \leq 10^5$), необходимо найти остаток от деления n-го числа Фибоначчи на m.

Схема решения. Можно найти n-ое число Фибоначчи и затем остаток от его деления на m. Если же мы опасаемся, что n-ое число Фибоначчи окажется слишком большим для представления в компьютере, можно воспользоваться свойствами операции «остаток от деления» и написать рекурсивную функцию $[F_n]_m = [[F_{n-1}]_m + [F_{n-2}]_m]_m$. Впрочем, рекурсия тоже требует много затрат, поэтому лучше заменить её на итерацию.

ЗАДАНИЕ ПО ИНФОРМАТИКЕ
ВАРИАНТ 73102 для 10 класса

Для заданий 1, 2, 4, 5 требуется разработать алгоритмы на языке блок-схем, псевдокоде или естественном языке

1. Для проверки, является ли большое целое простым, может использоваться вероятностный тест Ферма. Пусть $p > 2$ – проверяемое число. Тогда:
   - случайно выбираем a: $2 \le a \le p - 2$;
   - если $a^{p-1} \neq 1 \ \{\mathrm{mod}\ p\}$, то p – составное.

   В тесте Ферма эти проверки выполняются для t случайно выбираемых a.
   Написать алгоритм проверки вводимого числа на простоту по тесту Ферма.
   Примечание: $x = y \ \{\mathrm{mod}\ n\}$, если существует целое k, для которого $x = y + k \cdot n$.

Схема решения. В цикле t раз выбираем число a в заданном диапазоне. Для каждого a проверяем условие $a^{p-1} \neq 1 \ \{\mathrm{mod}\ p\}$. Для того чтобы при возведении в степень не получилось слишком большого числа (выходящего за пределы чисел, представимых в компьютере), можно применять операцию вычисления остатка от деления после каждого умножения на a. Если для какого-то значения a результат не равен 1, то число – составное, и проверку можно прекращать.

2. В пансионате для спортивного досуга детей оборудована специальная площадка с большим числом крупных клеток L, выложенных в дорожки одинаковой длины. По дорожкам (от начала до конца) с клетки на клетку любят прыгать отдыхающие дети. На каждой клетке нарисован вес – натуральное число. Выигрывает тот ребенок, который при прыжках набрал минимальный суммарный вес. В игре принимали участие $M \le L$ детей, прыгающих за один ход на 1, 2, 4 или 5 клеток. Предложите наиболее оптимальный способ обработки и хранения информации для моделирования ситуации (например, для определения победителя). Примечание: прыгать на 3 клетки нельзя.

Схема решения. Для хранения данных будем использовать матрицу из $L \times L$ элементов, в элементах которой записан вес каждой клетки. Результаты детей запишем в виде массива из $L \times M$ элементов, причём элемент равен 1, если ребёнок прыгал на данную клетку, и 0 – в противном случае. Также надо записать номер дорожки, по которой он прыгал. Чтобы получить итоговый результат, надо сложить веса клеток, на которые прыгал каждый ребёнок. Для проверки соблюдения правил будем вычислять разницу между индексами элементов в каждой строке второй матрицы, содержащих значение 1. Для этого положим сначала номер предыдущей использованной клетки 0, затем при нахождении в массиве значения 1 вычисляем разницу. Если она не равна 1, 3, 4 или 5, значит, правила была нарушены. Иначе прибавляем к общему результату вес текущей клетки и запоминаем номер текущей клетки как номер предыдущей.

3. Десятиклассник Сережа любит играть с калькулятором. Он часто сначала делит вещественные числа a и b друг на друга, а затем результат умножает на b. Выполнив эти действия много раз (сначала много делений, а затем столько же умножений), Сережа получил в результате некоторое число. Будет ли оно исходным? Объясните, почему?

Ответ. Не будет. Возникающая в процессе вычислений погрешность, обусловленная округлением вещественных чисел при представлении в ЭВМ, изменит результат.

4. Не используя дополнительный массив или простые методы сортировок, найти в матрице значения трех первых минимальных элементов.

Схема решения. Найдём минимальный элемент матрицы. Проверим, сколько таких элементов существует в матрице. Если их 3 и больше, значит, задача решена. Если нет, ещё раз найдём минимальный элемент из тех элементов, которые больше первого минимума. В случае необходимости найдём также третий минимальный элемент из тех элементов, которые больше первых двух минимумом.

5. Число трибоначчи – натуральное число, удовлетворяющее следующим соотношениям: $t_0 = 1,\ t_1 = 0,\ t_2 = 1,\ t_{n+3} = t_{n+2} + t_{n+1} + t_n,\ n \geq 2$. Даны целые числа n и m ($1 \leq n \leq 10^{18}$, $2 \leq m \leq 10^5$), необходимо найти остаток от деления n-го числа трибоначчи на m.

Схема решения. Можно найти n-ое число трибоначчи и затем остаток от его деления на m. Если же мы опасаемся, что n-ое число трибоначчи окажется слишком большим для представления в компьютере, можно воспользоваться свойствами операции «остаток от деления» и написать рекурсивную функцию $[t_{n+3}]_m = [[t_{n+2}]_m + [t_{n+1}]_m + [t_n]_m]_m$. Впрочем, рекурсия тоже требует много затрат, поэтому лучше заменить её на итерацию.