**Система оценивания экзаменационной работы по информатике и ИКТ**

**Ответы к заданиям с кратким ответом**

За правильный ответ на каждое задание части 1 ставится 1 балл. Если указаны два и более ответов (в том числе правильный), неверный ответ или ответ отсутствует - 0 баллов.

|  |  |  |  |
| --- | --- | --- | --- |
|  | № задания | Ответ |  |
|  | А1 | 4 |  |
|  | А2 | 2 |  |
|  | A3 | 1 |  |
|  | А4 | 3 |  |
|  | А5 | 4 |  |
|  | А6 | 3 |  |
|  | А7 | 1 |  |
|  | А8 | 1 |  |
|  | А9 | 1 |  |
|  | А10 | 2 |  |
|  | All | 3 |  |
|  | А12 | 2 |  |
|  | А13 | 4 |  |
|  |  | **Часть 2** |  |
| За правильный ответ на каждое задание части 2 ставится 1 балл , за неверный ответ или его отсутствие - 0 баллов. | | | |
|  | № задания | Ответ |  |
|  | В1 | 11212 |  |
|  | В2 | 10 |  |
|  | ВЗ | 5 |  |
|  | В4 | 48 |  |
|  | В5 | 9 |  |
|  | В6 | 120 |  |
|  | В7 | 15 |  |
|  | В8 | 37 |  |
|  | В9 | 13 |  |
|  | В10 | А122 |  |
|  | В11 | НСЕА |  |
|  | В12 | 2200 |  |
|  | В13 | 22 |  |
|  | В14 | 8 |  |
|  | В15 | 15 |  |

**Часть 3**

**КРИТЕРИИ ОЦЕНИВАНИЯ ЗАДАНИЙ С РАЗВЁРНУТЫМ ОТВЕТОМ**

Требовалось написать программу, при выполнении которой с клавиатуры считывается координата точки на прямой (х - действительное число) и определяется принадлежность этой точки одному из выделенных отрезков В и D (включая границы). Программист торопился и написал программу неправильно.

**C1**

|  |  |
| --- | --- |
| **Бейсик** | **Паскаль** |
| INPUT X  IF х>=-3 THEN  IF x<=9 THEN  IF X>1 THEN  PRINT "не принадлежит"  ELSE  PRINT "принадлежит"  ENDIF  ENDIF  ENDIF  END | var x: real;  begin  readln(x);  if x>=-3 then  if x<=9 then  if x>l then  write('не принадлежит')  else  write('принадлежит')  end. |
| **Си** | **Алгоритмический** |
| void main(void)  {  float x;  scanf("%f",&x);  if(x>=-3)  if (x< = 9)  if (x>l)  printf("He принадлежит");  else  printf("принадлежит");  } | алг  нач  вещ х  ввод X  если х>=-3 то  если х<=9 то  если х>1 то  вывод 'не принадлежит'  иначе  вывод 'принадлежит'  все все все  кон |

Последовательно выполните следующее.

1. Перерисуйте и заполните таблицу, которая показывает, как работает программа при аргументах, принадлежащих различным областям (A, B, C, D,E). Границы (точки -3, 1, 5 и 9) принадлежат заштрихованным областям (В и D соответственно).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Условие 1 | Условие 2 | Условие 3 | Программа выведет | Область |
| Область | (х>=-3) | (х <= 9) | (х>1) | о врабатывается верно |
| А |  |  |  |  |  |
| В |  |  |  |  |  |
| С |  |  |  |  |  |
| D |  |  |  |  |  |
| Е |  |  |  |  |  |

В столбцах условий укажите "да", если условие выполнится, "нет" если условие не выполнится, "—" (прочерк), если условие не будет проверяться, «не изв.», если программа ведет себя по-разному для разных значений, принадлежащих данной области. В столбце "Программа выведет" укажите, что программа выведет на экран. Если программа ничего не выводит, напишите "—" (прочерк). Если для разных значений, принадлежащих области, будут выведены разные тексты, напишите «не изв». В последнем столбце укажите "да" или "нет".

2. Укажите, как нужно доработать программу, чтобы не было случаев ее неправильной работы. (Это можно сделать несколькими способами, достаточно указать любой способ доработки исходной программы.)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Содержание верного ответа и указания по оцениванию**  (допускаются иные формулировки ответа, не искажающие его смысла) | | | | | |
| Элементы ответа: **1.** | | | | | |
| Область | Условие **1**  (х>=-3) | Условие 2  (х <=9) | Условие 3  (х>1) | Программа выведет | Область обрабатывается верно |
| А | Нет | — | — | — | Нет |
| В | Да | Да | Нет | Принадлежит | Да |
| С | Да | Да | Да | Не принадлежит | Да |
| D | Да | Да | Да | Не принадлежит | Нет |
| Е | Да | Нет | — | — | Нет |

|  |
| --- |
| 2. Возможная доработка (Паскаль): |
| if (х>=-3) and (х<=1) or (х>=5) and (х<=9) then |
| write('принадлежит') |
| else |
| write('не принадлежит') |
| Возможны и другие способы доработки. |
| Например: |
| if х>=-3 then |
| if х<=1 then |
| write('принадлежит') |
| else |
| if x>=5 then |
| if x<=9 then |
| write('принадлежит') |
| else |
| write('не принадлежит') |
| else |
| write('не принадлежит') |
| else |
| write('не принадлежит') |
| Другой пример: |
| if abs (abs(x-3)-4)<=2 then |
| write('принадлежит') |
| else |
| write('не принадлежит') |

|  |  |
| --- | --- |
| **Указания по оцениванию** | **Баллы** |
| Обратите внимание! В задаче требуется выполнить три действия.  1. Заполнить таблицу.  2. Исправить ошибку в условном операторе.  3. Исправить ошибку, связанную с неправильным набором условий. Баллы за данное задание начисляются как сумма баллов за верное выполнение каждого действия. Рассмотрим отдельно каждое действие.  1. Действие по заполнению таблицы считается выполненным, если в таблице нет ошибок или ошибки присутствуют только в одной строке.  2. Неправильное использование условного оператора, в результате чего при невыполнении первого или второго условия программа не выдавала ничего (отсутствуют случаи ELSE). Исправлением этой ошибки может быть либо добавление случая ELSE к каждому условию IF, либо объединение всех условий IF в одно при помощи конъюнкции.  В сложных случаях это действие считается выполненным, если программа выдает одно из двух сообщений: «принадлежит» или «не принадлежит» - для любых чисел *х*, при этом программа не стала работать хуже, чем раньше, т.е. для всех точек, для которых программа ранее выдавала верный ответ, доработанная программа также должна выдавать верный ответ.  3. Приведённых трёх ограничений недостаточно для описания двух областей (потеряно условие х>=5). Кроме того, необходимо учесть, что области не соединены. Исправлением этой ошибки может быть разбиение области на две части и использование дизъюнкции, либо использование сложной (для выведения) математической конструкции (||х-3|-4|<2), либо использование сложной комбинации каскадных условий.  В сложных случаях это действие считается выполненным, если верно определены заштрихованные области, т.е. программа выводит сообщение «принадлежит» для всех точек закрашенных областей, и только для них, для точек вне заштрихованных областей программа выводит «не принадлежит» или не выводит ничего. В работе (во фрагментах программ) допускается наличие отдельных синтаксических ошибок, не искажающих замысла автора решения |  |
| Выполнены все три действия | 3 |
| Правильно выполнены два действия из трёх (исправлены обе ошибки, но в пункте 1 задания не приведена таблица (либо таблица содержит ошибки в двух и более строках), либо приведена таблица (которая содержит ошибки не более чем в одной строке), но исправлена только одна ошибка программы).  При написании операций сравнения допускается одно неправильное использование строгих/нестрогих неравенств (считается несущественной ошибкой, погрешностью записи). Например, вместо «х>=5» используется «х>5» | 2 |
| Правильно выполнено только одно действие из трёх, т.е. либо только приведена таблица, которая содержит ошибки в не более чем одной строке, либо таблица не приведена (или приведена и содержит ошибки более чем в одной строке), но исправлена одна ошибка программы. При оценивании этого задания на 1 балл допускается не учитывать корректность работы программ на точках границ областей (вместо нестрогих неравенств в решении были использованы строгие неравенства или наоборот) | 1 |
| Все пункты задания выполнены неверно (таблица анализа правильности алгоритма не приведена, либо содержит ошибки более чем в двух строках, программа не приведена, либо ни одна из двух ошибок не исправлена). | 0 |
| *Максимальный балл* | 3 |

**C2**

Дан целочисленный массив из 30 элементов. Элементы массива могут принимать целые значения от 0 до 100. Опишите на русском языке или на одном из языков программирования алгоритм, позволяющий найти и вывести произведение элементов массива, которые имеют нечётное значение и делятся на 3. Гарантируется, что в исходном массиве есть хотя бы один элемент, значение которого нечётно и кратно 3.

Исходные данные объявлены так, как показано ниже. Запрещается использовать переменные, не описанные ниже, но разрешается не использовать часть из них. Исходные данные всегда подобраны так, что результат произведения не выходит за пределы объявленных типов данных.

|  |  |
| --- | --- |
| **Паскаль** | **Бейсик** |
| const  N=30;  var  a: array [1..N] of longint; i, j, p: longint;  begin  for i := 1 to N do readln(a[i]);  …end. | N=30  DIM A (N) AS LONG  DIM I, J, P AS LONG  FOR I = 1 TO N  INPUT A (I)  NEXT I  …  END |
| **СИ** | **Алгоритмический язык** |
| #include <stdio.h>  #define N 30  void main(void){  long a[N];  long i, j, p;  for (i=0; i<N; i++)  scanf("%ld" , &a[i]) ;  …  } | алг  нач  цел N=30  целтаб a[l:N]  цел i, j, р  нц для i от 1 до N  ввод a[i]  кц  …кон |

В качестве ответа вам необходимо привести фрагмент программы (или описание алгоритма на естественном языке), который должен находиться на месте многоточия. Вы можете записать решение также на другом языке программирования (укажите название и используемую версию языка программирования, например Borland Pascal 7.0) или в виде блок-схемы.

В этом случае вы должны использовать те же самые исходные данные и переменные, какие были предложены в условии (например, в образце, записанном на естественном языке).

|  |
| --- |
| **Содержание верного ответа и указания по оцениванию**  (допускаются иные формулировки ответа, не искажающие его смысла) |
| **На языке Паскаль** |
| p := 1;  for i := 1 to N do  if (a[i] mod 2<>0) and (a[i] mod 3=0) then  p := p \* a[i];  writeln(p); |
| **На Алгоритмическом языке** |
| p := 1  нц для i от 1 до N  если mod(a[i],2)<>0 и mod(a[i],3)=0  то  p := p \* a[i]  все  кц  вывод p |
| **На языке Бейсик** |
| P = 1  FOR I = 1 TO N  IF A(I) MOD 6=3 THEN  P = P \* A(I)  ENDIF  NEXT I  PRINT P |
| **На языке СИ** |
| p=1;  for (i=0; i<N; i++)  if (a[i]%2!=0 && a[i]%3==0)  p\*=a[i];  printf("%ld", p); |
| **На естественном языке** |
| Записываем в переменную P начальное значение, равное 1. В цикле от 1-го элемента до 30-го находим остаток от деления элемента исходного массива на 2 и на 3. Если остаток от деления на 2 не равен 0 и остаток от деления на 3 равен 0, то считаем произведение данного элемента массива и значения переменной P или достаточно проверить, что остаток от деления на 6 равен 3 (см. решение на языке Бейсик). Результат умножения сохраняем в переменную P. Переходим к следующему элементу массива. После завершения цикла выводим значение переменной P |

|  |  |
| --- | --- |
| **Указания по оцениванию** | **Баллы** |
| Предложен правильный алгоритм, выдающий верное значение. Допускается запись алгоритма на другом языке, использующая аналогичные переменные. В случае, если язык программирования использует типизированные переменные, описания переменных должны быть аналогичны описаниям переменных на естественном языке. Использование нетипизированных или необъявленных переменных возможно только в случае, если это допускается языком программирования, при этом количество переменных и их идентификаторы должны соответствовать условию задачи. В алгоритме, записанном на языке программирования, допускается наличие отдельных синтаксических ошибок, не искажающих замысла автора программы | 2 |
| В любом варианте решения может присутствовать не более одной ошибки из числа следующих.  1. Не инициализируется или неверно инициализируется переменная P (например, присваивается начальное значение, равное 0).  2. Неверно осуществляется проверка на нечётность.  3. На нечётность проверяется не значение элемента, а его индекс.  4. Неверно осуществляется проверка делимости на 3.  5. На делимость на 3 проверяется не значение элемента, а его индекс.  6. В сложном условии вместо логической операции «И» используется логическая операция «ИЛИ».  7. Неверно осуществляется накопление произведения в цикле (например, p := a[i]).  8. Отсутствует вывод ответа.  9. Используется переменная, не объявленная в разделе описания переменных.  10. Не указано или неверно указано условие завершения цикла.  11. Индексная переменная в цикле не меняется (например, в цикле while) или меняется неверно.  12. Неверно расставлены операторные скобки | 1 |
| Ошибок, перечисленных в п. 1–12, две или больше, или алгоритм сформулирован неверно. | 0 |
| *Максимальный балл* | 2 |

**C3**

Два игрока, Петя и Ваня, играют в следующую игру. Перед игроками лежит куча камней. Игроки ходят по очереди, первый ход делает Петя. За один ход игрок может добавить в кучу **один** камень или увеличить количество камней в куче в два раза. Например, имея кучу из 15 камней, за один ход можно получить кучу из 16 или 30 камней. У каждого игрока, чтобы делать ходы, есть неограниченное количество камней.

Игра завершается в тот момент, когда количество камней в куче становится не менее 22. Победителем считается игрок, сделавший последний ход, то есть первым получивший кучу, в которой будет 22 или больше камней. В начальный момент в куче было S камней, 1 < S < 21.

Будем говорить, что игрок имеет *выигрышную стратегию*, если он может выиграть при любых ходах противника. Описать стратегию игрока - значит описать, какой ход он должен сделать в любой ситуации, которая ему может встретиться при различной игре противника.

Выполните следующие задания. Во всех случаях обосновывайте свой ответ.

1. а) Укажите все такие значения числа S, при которых Петя может выиграть в один ход. Обоснуйте, что найдены все нужные значения S, и укажите выигрывающий ход для каждого указанного значения S.

б) Укажите такое значение S, при котором Петя не может выиграть за один ход, но при любом ходе Пети Ваня может выиграть своим первым ходом. Опишите выигрышную стратегию Вани.

2. Укажите два таких значения S, при которых у Пети есть выигрышная стратегия, причём

- Петя не может выиграть за один ход, и

- Петя может выиграть своим вторым ходом, независимо от того, как будет ходить Ваня.

Для каждого указанного значения S опишите выигрышную стратегию Пети.

3. Укажите значение S, при котором:

- у Вани есть выигрышная стратегия, позволяющая ему выиграть первым или вторым ходом при любой игре Пети, и

- у Вани нет стратегии, которая позволит ему гарантированно выиграть первым ходом.

Для указанного значения S опишите выигрышную стратегию Вани. Постройте дерево всех партий, возможных при этой выигрышной стратегии Вани (в виде рисунка или таблицы). На рёбрах дерева указывайте, кто делает ход, в узлах - количество камней в куче.

|  |
| --- |
| **Содержание верного ответа и указания к оцениванию**  (допускаются иные формулировки ответа, не искажающие его смысла) |
| 1. а) Петя может выиграть первым ходом, если S =11, …, 21. Во всех случаях нужно удвоить количество камней в куче. При меньших значениях S за один ход нельзя получить кучу, в которой больше 21 камня.  б) Ваня может выиграть первым ходом (как бы ни играл Петя), если исходно в куче будет S =10 камней. Тогда после первого хода Пети в куче будет 11 камней или 20 камней. В обоих случаях Ваня удваивает количество камней и выигрывает своим первым ходом.  2. Возможные значения S: 5 и 9. В этих случаях Петя, очевидно, не может выиграть первым ходом. Однако он может получить кучу из 10 камней. Эта позиция разобрана в п. 1б. В ней игрок, который будет ходить (теперь это Ваня), выиграть не может, а его противник (то есть,  Петя) следующим ходом выиграет.  3. Возможное значение S: 8. После первого хода Пети в куче будет 9 или 16 камней. Если в куче станет 16 камней, Ваня удвоит количество камней и выиграет первым ходом. Ситуация, когда в куче 9 камней, разобрана в п. 2. В этой ситуации игрок, который будет ходить (теперь это Ваня), выигрывает своим вторым ходом.  В таблице изображено дерево возможных партий при описанной стратегии Вани. Заключительные позиции (в них выигрывает Ваня) подчеркнуты. На рисунке это же дерево изображено в графическом виде (оба способа изображения дерева допустимы). |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Положения после очередных ходов | | | |
| И.п. | 1-й ход Пети (разобраны все ходы) | 1-й ход Вани (только ход по  стратегии) | 2-й ход Пети (разобраны все ходы) | 2-й ход Вани (только ход по стратегии) |
| 8 | 8+1 =9 | 9+1=10 | 10+1=11 | 11\*2=22 |
| 10\*2=20 | 20\*2=40 |
| 8\*2=16 | 16\*2=32 |  |  |



|  |  |
| --- | --- |
| **Указания по оцениванию** | **Баллы** |
| В задаче от ученика требуется выполнить 3 задания. Их трудность возрастает. Количество баллов в целом соответствует количеству выполненных заданий (подробнее см. ниже).  Ошибка в решении, не искажающая основного замысла, например, арифметическая ошибка при вычислении количества камней в заключительной позиции, при оценке решения не учитывается.  Первое задание считается выполненным полностью, если выполнены полностью оба пункта а) и б). Пункт а) считается выполненным полностью, если правильно указаны все позиции, в которых Петя выигрывает первым ходом и указано, каким должен быть первый ход. Пункт б) считается выполненным полностью, если правильно указана позиция, в которой Ваня выигрывает первым ходом и описана стратегия Вани, т.е. показано, как Ваня может получить кучу, в которой содержится нужное количество камней при любом ходе Пети.  Первое задание считается выполненным частично, если выполнены все следующие условия: (а) правильно указаны все позиции, в которых Петя выигрывает первым ходом, (б) правильно указана позиция, в которой Ваня выигрывает первым ходом, и явно сказано, что при любом ходе Пети Ваня может получить кучу, которая содержит нужное для выигрыша количество камней. Отличие от полного решения в том, что выигрышные ходы не указаны.  Второе задание выполнено, если правильно указаны обе позиции, выигрышные для Пети, и описаны соответствующие стратегии Пети - так, как это написано в примере решения, или другим способом, например, с помощью дерева всех партий, возможных при выбранных стратегиях.  Третье задание выполнено, если правильно указана позиция, выигрышная для Вани, и построено дерево всех партий, возможных при Ваниной стратегии. Должно быть явно сказано, что в этом дереве в каждой позиции, где должен ходить Петя, разобраны все возможные ходы, а для позиций, где должен ходить Ваня, - только ход, соответствующий стратегии, которую выбрал Ваня.  Во всех случаях стратегии могут быть описаны так, как это сделано в примере решения или другим способом |  |
| Выполнены второе и третье задания. Первое задание выполнено полностью или частично. Здесь и далее допускаются арифметические ошибки, которые не искажают сути решения и не приводят к неправильному ответу (см. выше) | 3 |
| Не выполнены условия, позволяющие поставить 3 балла, выполнено одно из следующих условии.  1. Задание 3 выполнено полностью.  2. Первое и второе задания выполнены полностью.  3. Первое задание выполнено полностью или частично; для заданий 2 и 3 указаны правильные значения S | 2 |
| Не выполнены условия, позволяющие поставить 3 или 2 балла, выполнено одно из следующих условии.  1. Первое задание выполнено полностью.  2. Во втором задании правильно указано одно из двух возможных значений S и для этого значения указана и обоснована выигрышная стратегия Пети.  3. Первое задание выполнено частично и для одного из остальных заданий правильно указано значение S.  4. Для второго и третьего задания правильно указаны значения S | 1 |
| Не выполнено ни одно из условий, позволяющих поставить 3, 2 или 1 балл | 0 |
| *Максимальный балл* | 3 |

**C4**

На вход программе подаются сведения о пассажирах, желающих сдать свой багаж в камеру хранения на заранее известное время до полуночи. В первой строке сообщается число пассажиров N, которое не меньше 3, но не превосходит 1000; во второй строке - количество ячеек в камере хранения К, которое не меньше 10, но не превосходит 1000. Каждая из следующих N строк имеет следующий формат:

<Фамилия> <время сдачи багажа> <время освобождения ячейки>, где <Фамилия> - строка, состоящая не более чем из 20 непробельных символов; <время сдачи багажа> - через двоеточие два целых числа, соответствующие часам (от 00 до 23 - ровно 2 символа) и минутам (от 00 до 59 - ровно 2 символа); <время освобождения ячейки> имеет тот же формат. <Фамилия> и <время сдачи багажа>, а также <время сдачи багажа> и <время освобождения ячейки> разделены одним пробелом. Время освобождения больше времени сдачи.

Сведения отсортированы в порядке времени сдачи багажа. Каждому из пассажиров в камере хранения выделяется свободная ячейка с минимальным номером. Если в момент сдачи багажа свободных ячеек нет, то пассажир уходит, не дожидаясь освобождения одной из них.

Требуется написать программу (укажите используемую версию языка программирования, например Borland Pascal 7.0), которая будет выводить на экран для каждого пассажира номер ему предоставленной ячейки (можно сразу после ввода данных очередного пассажира). Если ячейка пассажиру не предоставлена, то его фамилия не печатается.

Пример входных данных:

3

10

Иванов 09:45 12:00

Петров 10:00 11:00

Сидоров 12:00 13:12

Результат работы программы на этих входных данных: Иванов 1

Петров 2

Сидоров 1

|  |
| --- |
| **Содержание верного ответа и указания по оцениванию**  (допускаются иные формулировки ответа, не искажающие его смысла) |
| Программа верно читает входные данные, сразу запоминая только время окончания хранения багажа в массиве, соответствующем ячейкам камеры хранения. Подходящая ячейка определяется путём последовательного просмотра элементов этого массива до первого свободного или такого, в котором записано время окончания хранения, не превосходящее текущего времени сдачи очередного багажа. В случае удачного выбора ячейки фамилия и номер ячейки распечатываются. Баллы начисляются только за программу, которая решает задачу хотя бы для частного случая. Время можно как переводить в минуты, так и хранить в виде строки, сравнивая затем строки непосредственно. В последнем случае упрощается ввод данных |
| **Пример правильной и эффективной программы на языке Паскаль:** |
| var p:array[1..1000] of integer;  c,c1:char;  i,j,N,K:integer;  name:string;  time1,time2:integer;  begin  readln(N,K);  for i:=1 to K do  p[i]:=0;  for i:=1 to N do  begin  name:='';  repeat  read(c);  name:=name+c  until c=' '; {считана фамилия}  read(c,c1); {считаны часы первого времени}  time1:=60\*((ord(c)-ord('0'))\*10+ ord(c1)-ord('0'));  read(c,c,c1); {пропущено двоеточие, и считаны минуты}  time1:=time1+(ord(c)-ord('0'))\*10+ord(c1)-ord('0');  read(с,c,c1); {считаны часы второго времени}  time2:=60\*((ord(c)-ord('0'))\*10+ ord(c1)-ord('0'));  readln(c,c,c1); {пропущено двоеточие, и считаны минуты}  time2:=time2+(ord(c)-ord('0'))\*10+ord(c1)-ord('0');  for j:=1 to K do  if p[j]<=time1 then  begin  p[j]:=time2;  writeln(name,' ',j);  break;  end;  end;  end. |
| **Пример правильной и эффективной программы на языке Бейсик:** |
| DIM p(1000) AS INTEGER  DIM s AS STRING  DIM nm AS STRING  INPUT n  INPUT k  FOR i = 1 TO k  p(i) = 0  NEXT i  FOR j = 1 TO n  LINE INPUT s  c$ = MID$(s, 1, 1)  i = 1  WHILE NOT (c$ = " ")  i = i + 1  c$ = MID$(s, i, 1)  WEND  nm = MID$(s, 1, i)  time1 = (ASC(MID$(s, i + 1, 1)) - ASC("0")) \* 60 \* 10  time1 = time1 + (ASC(MID$(s, i + 2, 1)) - ASC("0")) \* 60  time1 = time1 + (ASC(MID$(s, i + 4, 1)) - ASC("0")) \* 10  time1 = time1 + (ASC(MID$(s, i + 5, 1)) - ASC("0"))  time2 = (ASC(MID$(s, i + 7, 1)) - ASC("0")) \* 60 \* 10  time2 = time2 + (ASC(MID$(s, i + 8, 1)) - ASC("0")) \* 60  time2 = time2 + (ASC(MID$(s, i + 10, 1)) - ASC("0")) \* 10  time2 = time2 + (ASC(MID$(s, i + 11, 1)) - ASC("0"))  FOR i = 1 TO k  IF time1 >= p(i) THEN  p(i) = time2  PRINT nm, i  GOTO 10  ENDIF  NEXT i  10 NEXT j  END |

|  |  |
| --- | --- |
| **Указания по оцениванию** | **Баллы** |
| Программа работает верно, т.е. корректно выделяет из входных данных время, ищет первую свободную ячейку и распечатывает в случае успеха результат. Фамилии пассажиров и время сдачи багажа при этом не запоминаются. Допускается наличие в тексте программы одной синтаксической ошибки: пропущен или неверно указан знак пунктуации; неверно написано или пропущено зарезервированное слово языка программирования; не описана или неверно описана переменная; применяется операция, не допустимая для соответствующего типа данных. Допускается наличие одной из нерациональностей: сохраняются фамилии пассажиров; сохраняется как время сдачи багажа, так и время освобождения ячейки; до анализа данных очередного пассажира анализируется массив, соответствующий ячейкам камеры хранения на предмет освобождения ячеек к данному времени | 4 |
| Программа работает в целом верно, но содержит, по крайней мере, две из указанных выше нерациональностей; допускается наличие до трёх синтаксических ошибок, описанных выше | 3 |
| Программа работает в целом верно, но не всегда определяет для хранения допустимую ячейку с минимальным номером или некорректно работает в случае отсутствия свободных ячеек. Возможно, в реализации алгоритма содержится одна-две ошибки (используется знак «<» вместо «>», «or» вместо «and» и т.п.). Возможно, некорректно организовано считывание входных данных. Допускается до трёх ошибок в ходе решения задачи. Допускается наличие до пяти синтаксических ошибок, описанных выше | 2 |
| Программа неверно работает при некоторых входных данных и, возможно, содержит ошибку в определении свободной ячейки. Допускается до четырёх различных ошибок в ходе решения задачи, в том числе описанных в критериях. Допускается наличие до семи синтаксических ошибок, описанных выше | 1 |
| Задание выполнено неверно | 0 |
| *Максимальный балл* | 4 |