МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Уральский государственный экономический университет»
(УрГЭУ)

УТВЕРЖДАЮ:

Председатель приемной комиссии

_____Я.П. Силин

ПРОГРАММА
вступительных испытаний
по предмету
«Основы программирования»

для поступающих на обучение по программам бакалавриата
на базе среднего профессионального образования

Екатеринбург

Программа по предмету «Основы программирования» составлена в соответствии с приказом Министерства образования РФ от 05.03.2004 № 1089 «Об утверждении федерального компонента государственных образовательных стандартов начального общего, основного общего и среднего (полного) общего образования» (с изменениями внесенными приказами Министерства образования и науки от 03.06.2008 № 164, от 31.08.2009 № 320, от 19.10.2009 № 427, от 21.01.2012 № 39, от 31.01.2012 №69) и рабочей программы учебной дисциплины, разработанной на основе ФГОС по специальности среднего профессионального образования 09.02.03 и примерной программы, рекомендованной ФГАУ ФИРО, заключение Экспертного совета № 092 от «02» марта 2012 г.

Вступительное испытание проводится в форме тестирования. На выполнение экзаменационной работы отводится 1,5 часа (90 минут).

ТРЕБОВАНИЯ К УРОВНЮ ПОДГОТОВКИ АБИТУРИЕНТОВ

В ходе вступительного испытания абитуриент должен продемонстрировать:

- знания следующего фундаментального теоретического материала:
  - основные понятия и законы математической логики;
  - понятие алгоритма, его свойств, способов записи;
  - основные алгоритмические конструкции;
  - основные конструкции языка программирования;
  - основные элементы программирования.
- следующие умения и навыки:
  - строить таблицы истинности и логические схемы;
  - владеть навыками алгоритмического мышления и понимание необходимости формального описания алгоритмов;
  - формально исполнять алгоритмы, записанные на естественных и алгоритмических языках, в том числе на языках программирования;
  - анализировать текст программы с точки зрения соответствия записанного алгоритма поставленной задаче и изменять его в соответствии с заданием;
  - уметь анализировать алгоритм, содержащий ветвление и цикл;
  - уметь анализировать алгоритм логической игры;
  - уметь построить дерево игры по заданному алгоритму и найти выигрышную стратегию;
  - уметь исполнить рекурсивный алгоритм;
  - уметь вычислить рекуррентные выражения;
  - использовать стандартные алгоритмические конструкции при программировании;

- уметь составить алгоритм обработки числовой последовательности и записать его в виде простой программы (10–15 строк) на языке программирования;
- уметь обрабатывать целочисленную информацию с использованием сортировки;
- уметь создавать собственные программы (20–40 строк) для анализа числовых последовательностей;
- реализовывать сложный алгоритм с использованием современных систем программирования.

СТРУКТУРА ВСТУПИТЕЛЬНОГО ИСПЫТАНИЯ, КРИТЕРИИ ОЦЕНИВАНИЯ

Каждый вариант экзаменационной работы состоит из двух частей и включает в себя 23 задания, различающихся формой и уровнем сложности.
Часть 1 содержит 22 задания с кратким ответом.
В экзаменационной работе предложены следующие разновидности заданий с кратким ответом:
–задания на вычисление определенной величины;
–задания на установление правильной последовательности, представленной в виде строки символов по определенному алгоритму.
Ответ на задания части 1 дается соответствующей записью в виде натурального числа или последовательности символов (букв или цифр), записанных без пробелов и других разделителей.
Часть 2 содержит 1 задание с кратким ответом.
Правильное выполнение заданий части 1 оценивается в 4 балла.
Максимальное количество баллов, которое можно получить за выполнение заданий части 1 – 88. Выполнение задания 23 оценивается в 12 баллов.
Максимальное количество баллов, которое может получить абитуриент по итогам вступительного испытания, равняется 100 баллам.

СОДЕРЖАНИЕ РАЗДЕЛОВ ПРОГРАММЫ

Логика и алгоритмы
Высказывания, логические операции, кванторы, истинность высказывания. Операции «импликация», «эквивалентность». Примеры законов алгебры логики. Эквивалентные преобразования логических выражений. Построение логического выражения с данной таблицей истинности. Логические функции.

Цепочки (конечные последовательности), деревья, списки, графы, массивы. Решение алгоритмических задач, связанных с анализом графов (примеры: построение оптимального пути между вершинами ориентированного ациклического графа; определение количества различных путей между вершинами). Использование графов, деревьев, списков при описании объектов и процессов окружающего мира. Решение алгоритмических задач, связанных с анализом графов (примеры: построение оптимального пути между вершинами ориентированного графа; определение количества различных путей между вершинами). Использование деревьев при решении алгоритмических задач (примеры: анализ работы рекурсивных алгоритмов, разбор арифметических и логических выражений). Дискретные игры двух игроков с полной информацией. Выигрышные стратегии.
Индуктивное определение объектов. Рекурсивные алгоритмы.
Кодирование с исправлением ошибок. Коды с возможностью обнаружения и исправления ошибок.
Сортировка. Постановка задачи сортировки Сортировка одномерных массивов. Квадратичные алгоритмы сортировки (пример: сортировка пузырьком).

Элементы теории алгоритмов
Основные понятия, связанные с использованием основных алгоритмических конструкций.
Построение алгоритмов и практические вычисления. Исполнение и анализ отдельных алгоритмов, записанных в виде блок-схемы, на алгоритмическом языке или на языках программирования. Подпрограммы. Табличные величины (массивы). Определение возможных результатов работы алгоритмов управления исполнителями и вычислительных алгоритмов. Определение исходных данных, при которых алгоритм может дать требуемый результат. Метод динамического программирования. Анализ алгоритмов: определение входных данных, при которых алгоритм даёт указанный результат; определение результата алгоритма без его полного пошагового выполнения.
Составление алгоритмов для конкретного исполнителя.

Технологии программирования
Типы данных. Типы и структуры данных. Кодирование базовых алгоритмических конструкций на выбранном языке программирования. Логические переменные. Символьные и строковые переменные. Операции над строками. Двумерные массивы (матрицы).
Основные конструкции языка программирования. Система программирования Операторы языка программирования, основные конструкции языка

программирования. Подробное знакомство с одним из универсальных процедурных языков программирования. Запись алгоритмических конструкций и структур данных в выбранном языке программирования. Обзор процедурных языков программирования. Подпрограммы (процедуры, функции). Параметры подпрограмм. Рекурсивные процедуры и функции.

Основные этапы разработки программ. Разбиение задачи на подзадачи. Этапы решения задач на компьютере. Интегрированная среда разработки программ на выбранном языке программирования. Интерфейс выбранной среды. Составление алгоритмов и программ в выбранной среде программирования. Приёмы отладки программ. Проверка работоспособности программ с использованием трассировочных таблиц Разработка и программная реализация алгоритмов решения типовых задач базового уровня из различных предметных областей Структурное программирование. Проверка условия выполнения цикла до начала выполнения тела цикла и после выполнения тела цикла: постусловие и пред условие цикла. Разработка программ, использующих подпрограммы. Понятие об объектно-ориентированном программировании. Объекты и классы. Использование модулей (компонентов) при разработке программ.

Примерные задания

1. Запишите число, которое будет напечатано в результате выполнения программы. Для Вашего удобства программа представлена на пяти языках программирования.

Бейсик	**Python**
DIM S, N AS INTEGER S = 33 N = 1 WHILE S > 0 S = S – 7 N = N * 3 WEND PRINT(N)	s = 33 n = 1 while s > 0: s = s - 7 n = n * 3 print(n)
Паскаль	**Алгоритмический язык**

var s, n: integer;
begin
s := 33;
n := 1;
while s > 0 do
begin
s := s – 7;
n := n * 3
end;
writeln(n)
end.
	нач
	цел s, n
	s := 33
	n := 1
	нц пока s > 0
	s := s – 7
	n := n * 3
	кц
	вывод n
	кон

Си	

```
#include
int main(void)
{ int s, n;
s = 33;
n = 1;
while (s > 0) {
s = s – 7;
n = n * 3;
}
printf("%d\n", n);
}
```

2

. Исполнитель Чертёжник перемещается на координатной плоскости, оставляя след в виде линии. Чертёжник может выполнять команду **Сместиться на (a, b)** (где *a, b* – целые числа), перемещающую Чертёжника из точки с координатами *(x, y)*, в точку с координатами *(x+a, y+b)*. Если числа *a, b* положительные, значение соответствующей координаты увеличивается, если отрицательные — уменьшается.

Какую команду надо выполнить Чертёжнику, чтобы вернуться в исходную точку, из которой он начал движение?

1) Сместиться на (4, –2)
2) Сместиться на (–4, 2)
3) Сместиться на (2, –8)
4) Сместиться на (–2, 8)

3. Алгоритм вычисления значения функции $F(n)$, где n – *натуральное* число, задан следующими соотношениями:

$F(n) = 1$ при $n = 1$;

$F(n) = n + F(n - 1)$, если n чётно,

$F(n) = 2 \times F(n - 2)$, если $n > 1$ и при этом n нечётно.

Чему равно значение функции $F(26)$?

4. Исполнитель Редактор получает на вход строку цифр и преобразовывает её. Редактор может выполнять две команды, в обеих командах *v* и *w* обозначают цепочки цифр.

А) **заменить** (*v, w*).

Эта команда заменяет в строке первое слева вхождение цепочки *v* на цепочку *w*.

Например, выполнение команды

заменить (111, 27)

преобразует строку 05111150 в строку 0527150.

Если в строке нет вхождений цепочки *v*, то выполнение команды

заменить (*v, w*) не меняет эту строку.

Б) **нашлось** (*v*).

Эта команда проверяет, встречается ли цепочка *v* в строке исполнителя Редактор. Если она встречается, то команда возвращает логическое значение «истина», в

противном случае возвращает значение «ложь». Строка исполнителя при этом не изменяется.
Цикл
ПОКА *условие*
последовательность команд
КОНЕЦ ПОКА
выполняется, пока условие истинно.
В конструкции
ЕСЛИ *условие*
ТО *команда1*
ИНАЧЕ *команда2*
КОНЕЦ ЕСЛИ
выполняется *команда1* (если условие истинно) или *команда2* (если условие ложно).
Какая строка получится в результате применения приведённой ниже программы к строке, состоящей из 70 идущих подряд цифр 8? В ответе запишите полученную строку.
НАЧАЛО
ПОКА **нашлось** (2222) ИЛИ **нашлось** (8888)
ЕСЛИ **нашлось** (2222)
ТО **заменить** (2222, 88)
ИНАЧЕ **заменить** (8888, 22)
КОНЕЦ ЕСЛИ
КОНЕЦ ПОКА
КОНЕЦ

5. Требовалось написать программу, при выполнении которой с клавиатуры считывается положительное целое число N, не превосходящее 10^9, и определяется сумма цифр этого числа. Программист торопился и написал программу неправильно.

Бейсик	**Python**
DIM N AS LONG INPUT N sum = 1 WHILE N > 0 D = N MOD 10 N = N \ 10 sum = sum + 1 WEND PRINT sum END	N = int(input()) sum = 1 while N > 0: d = N%10 N = N // 10 sum = sum + 1 print(sum)
Паскаль	**Алгоритмический язык**

var N: longint;
sum, d: integer;
begin
readln(N);
	алг
	нач
	цел N, d, sum
	ввод N

```
sum := 1;
while N > 0 do
begin
d := N mod 10;
    N := N div 10;
sum := sum + 1;
end;
writeln(sum);
end.
```

```
sum := 1
нц пока N > 0
d := mod(N, 10)
    N := div(N, 10)
sum := sum + 1
кц
вывод sum
кон
```

Си

```
#include
int main() {
long int N;
int sum, d;
scanf("%ld", &N);
sum = 1;
while (N > 0) {
d = N%10;
    N = N / 10;
sum = sum + 1;
}
printf("%d", sum);
return 0;
}
```

Последовательно выполните следующее.

1. Напишите, что выведет эта программа при вводе числа 256.

2. Приведите пример такого трёхзначного числа, при вводе которого программа выдаёт правильный результат.

3. Найдите все ошибки в этой программе (их может быть одна или несколько). Для каждой ошибки:

1) выпишите строку, в которой сделана ошибка;

2) укажите, как исправить ошибку, т. е. приведите правильный вариант строки.

Достаточно указать ошибки и способ их исправления для одного языка программирования. Обратите внимание, что требуется найти ошибки в имеющейся программе, а не написать свою, возможно, использующую другой алгоритм решения. Исправление ошибки должно затрагивать только строку, в которой находится ошибка.

СПИСОК ЛИТЕРАТУРЫ, РЕКОМЕНДУЕМОЙ ДЛЯ ПОДГОТОВКИ

1. Ушаков Д.М. ЕГЭ-2022. Информатика. 10 тренировочных вариантов экзаменационных работ для подготовки к ЕГЭ. — М.: АСТ, 2021.
2. Ушаков Д.М. ЕГЭ-2022. Информатика. 20 тренировочных вариантов экзаменационных работ для подготовки к ЕГЭ. — М.: АСТ, 2021.
3. Крылов С.С., Чуркина Т.Е. ЕГЭ 2022. Информатика и ИКТ. Типовые экзаменационные варианты: 10 вариантов. — М.: Национальное образование, 2021.
4. Лещинер В.Р. ЕГЭ 2022. Информатика. ТВЭЗ. 16 вариантов. — М.: Экзамен, 2022.

Председатель экзаменационной комиссии
по предмету «Основы программирования» Д.М. Назаров