Документ подписан простой электронной подписью
Информация о владельце:
ФИО: Кислова Наталья Николаевна
Должность: Проректор по УМР и качеству образования
Дата подписания: 03.09.2024 12:49:48
Уникальный программный ключ:
52802513f5b14a975b3e9b13008093d5726b159bf6064f865ae65b96a966c035

МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Самарский государственный социально-педагогический университет»
Информатики, прикладной математики и методики их преподавания

Утверждаю
Проректор по учебно-методической
работе и качеству образования
_____ Н.Н. Кислова

Бурцев Николай Павлович

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ
для проведения промежуточной аттестации по дисциплине
«Введение в процедурное программирование»

Направление подготовки:
44.03.05 Педагогическое образование (с двумя профилями подготовки)
Направленность (профиль):
«Информатика» и «Дополнительное образование (в области информатики и ИКТ)»

Квалификация выпускника
Бакалавр

Рассмотрено
Протокол №11 от 25.06.2024г.
Заседания кафедры информатики, прикладной математики и методики их преподавания

Одобрено
Начальник Управления образовательных программ
Н.А. Доманина

Самара 2024

Пояснительная записка

Фонд оценочных средств (далее – ФОС) для промежуточной аттестации по дисциплине «Введение в процедурное программирование» разработан в соответствии с требованиями Федерального государственного образовательного стандарта высшего образования – бакалавриат по направлению подготовки 44.03.05 Педагогическое образование (с двумя профилями подготовки), утвержденного приказом Министерства образования и науки Российской Федерации от 22 февраля 2018 г. № 125 (зарегистрирован Министерством юстиции Российской Федерации 15 марта 2018 г., регистрационный № 50358), с изменениями, внесенными приказами Министерства науки и высшего образования Российской Федерации от 26 ноября 2020 г. № 1456 (зарегистрирован Министерством юстиции Российской Федерации 27 мая 2021 г., регистрационный № 63650) и от 8 февраля 2021 г. № 83 (зарегистрирован Министерством юстиции Российской Федерации 12 марта 2021 г., регистрационный № 62739), основной профессиональной образовательной программой «Информатика» и «Дополнительное образование (в области информатики и ИКТ)» с учетом требований профессионального стандарта «01.001 Педагог (педагогическая деятельность в сфере дошкольного, начального общего, основного общего, среднего общего образования) (воспитатель, учитель)», утвержденного приказом Министерства труда и социальной защиты Российской Федерации от 18 октября 2013 г. № 544н. (зарегистрирован Министерством юстиции Российской Федерации 6 декабря 2013 г., регистрационный № 30550), с изменениями, внесенными приказами Министерства труда и социальной защиты Российской Федерации от 25 декабря 2014 г. № 1115н (зарегистрирован Министерством юстиции Российской Федерации 19 февраля 2015 г., регистрационный № 36091) и от 5 августа 2016 г. № 422н (зарегистрирован Министерством юстиции Российской Федерации 23 августа 2016 г., регистрационный № 43326), 01.003 «Педагог дополнительного образования детей и взрослых» утвержденный приказом Министерства труда и социальной защиты Российской Федерации от 22 сентября 2012 г. № 652н от 22.09.2021 г. (Зарегистрировано в Минюсте России 17.12.2021 N 66403).

Цель ФОС для промежуточной аттестации – установление уровня сформированности компетенций УК-1.

Задачи ФОС для промежуточной аттестации - контроль качества и уровня достижения результатов обучения по формируемым в соответствии с учебным планом компетенциям:

УК-1. Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач

УК-1.1. Анализирует задачу, выделяя этапы ее решения, действия по решению задачи

Знает: этапы решения задачи на компьютере.

Умеет: анализировать задачу, выделяя её базовые составляющие; осуществлять декомпозицию задачи.

УК-1.2. Находит, критически анализирует и выбирает информацию, необходимую для решения поставленной задачи

Знает: способы формализации алгоритмов на языках процедурного программирования.

Умеет: осуществлять постановку задачи; анализировать условие и определять оптимальный метод решения поставленной задачи.

УК-1.3. Рассматривает различные варианты решения задачи, оценивает их преимущества и риски

Знает: методы процедурного программирования; основные конструкции языков программирования; основные типы данных и операторы; приемы оптимизации алгоритмов по памяти и времени.

Умеет: строить математическую модель; составлять алгоритм решения задачи; записывать разрабатываемые алгоритмы на процедурных языках программирования.

УК-1.4. Грамотно, логично, аргументированно формирует собственные суждения и оценки; отличает факты от мнений, интерпретаций, оценок в рассуждениях других участников деятельности

Знает: основные виды ошибок, возникающих при решении задачи.

Умеет: комментировать синтаксические и семантические ошибки, возникающие при некорректном выполнении программы; отлаживать и тестировать задачи; составлять систему тестов для автоматизированной проверки корректности программы.

УК-1.5. Определяет и оценивает практические последствия возможных вариантов решения задачи

Умеет: выполнять оценку сложности алгоритмов, проводить анализ и оценивание полученных результатов.

Требование к процедуре оценки:

Помещение: особых требований нет

Расходные материалы: лист бумаги, ручка

Доступ к дополнительным справочным материалам: не предусмотрен

Нормы времени: 130 мин

Комплект оценочных средств для проведения промежуточной аттестации 1 семестр

Раздел «Процедурное программирование»

Проверяемая компетенция:

Универсальная компетенция УК-1.

Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач.

Проверяемый индикатор достижения компетенции:
УК-1.1: анализирует задачу, выделяя этапы ее решения, действия по решению задачи.

Проверяемые результаты обучения:
Знает: этапы решения задачи на компьютере.
Умеет: анализировать задачу, выделяя её базовые составляющие; осуществлять декомпозицию задачи

Проверяемый индикатор достижения компетенции:
УК-1.2 Находит, критически анализирует и выбирает информацию, необходимую для решения

Проверяемые результаты обучения:
Знает: способы формализации алгоритмов на языках программирования высокого уровня.
Умеет: осуществлять постановку задачи; анализировать условие и определять оптимальный метод решения поставленной задачи.

Проверяемый индикатор достижения компетенции:
УК-1.3. Рассматривает различные варианты решения задачи, оценивает их преимущества и риски

Проверяемые результаты обучения:
Знает: методы процедурного, объектно-ориентированного, функционального и визуального программирования; основные конструкции языков программирования; основные типы данных и операторы; приемы оптимизации алгоритмов по памяти и времени.
Умеет: строить математическую модель; составлять алгоритм решения задачи; записывать разрабатываемые алгоритмы на языках программирования высокого уровня.

Проверяемый индикатор достижения компетенции:
УК-1.4. Грамотно, логично, аргументированно формирует собственные суждения и оценки; отличает факты от мнений, интерпретаций, оценок в рассуждениях других участников деятельности.

Знает: основные виды ошибок, возникающих при решении задачи.
Умеет: комментировать синтаксические и семантические ошибки, возникающие при некорректном выполнении программы; отлаживать и тестировать задачи; составлять систему тестов для автоматизированной проверки корректности программы.

Проверяемый индикатор достижения компетенции:
УК-1.5. Определяет и оценивает практические последствия возможных вариантов решения задачи

Проверяемые результаты обучения:
Умеет: выполнять оценку сложности алгоритмов, проводить анализ и оценивание полученных результатов.

Тип (форма) задания: тестовые задания с выбором одного или нескольких вариантов ответа (A), практические задания (B, C).

Содержание задания:

A1. Какое высказывание наиболее точно определяет понятие «модель»
a) точная копия оригинала;
b) оригинал в миниатюре;
c) образ оригинала с наиболее присущими свойствами;
d) начальный замысел будущего объекта?

A2. Какой этап решения задач на ЭВМ непосредственно предшествует этапу программирования
a) постановка задачи;
b) анализ и исследование задачи, модели;
c) разработка алгоритма;
d) тестирование и отладка?

A3. Запись алгоритма на языке конкретного исполнителя – это...
a) алгоритм;
b) команда;
c) программа;
d) процедура.

A4. Фактические параметры процедуры:
a) описываются в ее заголовке;

b) указываются при описании данных в программе;
c) перечисляются при ее вызове;
d) нигде не указываются.

A5. Сколько раз будут выполнены операторы тела цикла при выполнении следующего фрагмента программы:
A:=1; N:=0; S:=0;
While A>1/1050 Do
Begin A:=Exp(–N*Ln(2)): S:=S+A: N:=N+1 End;
a) 11;
b) 1050;
c) 10;
d) 100?

A6. При истинности какого условия последовательность переменных A, B, C не является упорядоченной по возрастанию:
a) (A<B) AND (NOT(B>=C));
b) NOT ((A>B) OR (B>C));
c) (A<=B) AND (NOT(B>=C));
d) NOT ((A<=B) AND (B<=C))?

A7. Какое значение будет иметь переменная d после выполнения операторов
d = 0;
If a>b then begin d:=b; d:=d+a end else; d:=d*10;
при a = 1; b = 3:
a) 0;
b) 4;
c) 10;
d) 30?

A8. Какое значение будет иметь переменная d после выполнения операторов при k=1: d:=1;
case k mod 10 of
2,3,5: d:=2;
1,4: d:=3;
7..9: d:=4
End;
a) 1;
b) 2;
c) 3;
d) 4?

A9. Какое значение будет иметь переменная y после выполнения следующих фрагментов программы:
var y: real; k: integer;
y:=1; for k:=1 to 3 do y:=y+k; y:=y*10;
a) 70;
b) 2230;
c) 30;
d) 300?

A10. Какое значение будет иметь переменная y после выполнения следующих фрагментов программы:
Var y: real; k: integer;
y:=0; k:=1; repeat y:=y+1/k; k:=k-1 until k<=1;
a) 1;
b) 0;
c) -1;
d) 0.5?

A11. Служебное слово VAR в программе на языке Паскаль фиксирует начало раздела программы, содержащего:
a) операторы;
b) список меток;
c) перечень констант;
d) описание переменных.

A12. В алфавит языка Паскаль не входит служебное слово:
a) THEN;

b) BEGIN;
c) END;
d) STEP.

A13. Комментарий к тексту программы на языке Паскаль заключается:
a) в фигурные скобки;
b) в круглые скобки;
c) в квадратные скобки;
d) в апострофы.

A14. В качестве имени в языке Паскаль можно использовать
a) 1r;
b) or;
c) перем;
d) sum.

A15. Операторы в программе на языке Паскаль отделяются друг от друга:
a) двоеточием;
b) пробелом;
c) запятой;
d) точкой с запятой.

A16. Чему равен результат выражения 3 – 8 + 21 div 3
a) 2;
b) 5;
c) 12;
d) 18 ?

A17. Какое из приведенных неравенств верно:
a) –4.9876543234 E–02<–0,03;
b) –4.9876543234 E–02<–0,5;
c) –4.9876543234 E–02>–0,03;
d) –4.9876543234 E–02<–0,4?

A18. Какая из перечисленных операций не относится к логическим:
a) OR;
b) MOD;
c) NOT;
d) AND?

A19. Определите, какой элемент пропущен в программе, предназначенной для подсчета числа делителей натурального числа N.

```
Var res, I, N: integer;
Begin  Readln(N);  res:=0;  for I:=1 to N do  if … then res:=res+1;  writeln(res); end.
```

a) N div 2=0;
b) I mod 2<>0;
c) N mod I=0;
d) N div I=1.

A20. Определите, какой элемент пропущен в программе, предназначенной для подсчета числа делителей натурального числа N.

```
Var res, I, N: integer;
Begin  Readln(N);  res:=0;  for I:=1 to N do  if … then res:=res+1;  writeln(res); end.
```

a) N div 2=0
b) I mod 2<>0;
c) N mod I=0;
d) N div I=1.

Оценочный лист к типовому заданию А

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
c	c	c	c	a	d	a	c	a	a
A11	A12	A13	A14	A15	A16	A17	A18	A19	A20

d	d	a	d	d	a	a	b	c	c

B1. Определите, что будет напечатано в результате работы следующего фрагмента программы:

```
алг
нач
   цел k, s
   s := 5
   k := 0
   нц пока k < 15
      k := k + 2
      s := s + k
   кц
   вывод s
кон
```

B2. Определите, какое число будет напечатано в результате выполнения следующего алгоритма.

```
var a,b,t,M,R: integer;
   Function F (x:integer): integer;
      begin
         F := x*x + 6*x + 10
      end;
begin
   a := -10; b := 10;
   M := a; R := F(a);
   for t := a to b do begin
      if (F(t) < R) then begin
         M := t;
         R := F(t)
      end
   end;
   write(M)
end.
```

B3. Ниже на языке Паскаль записаны две рекурсивные функции F и G. Чему будет равно значение, вычисленное при выполнении вызова F(6)?

```
function F(n: integer): integer;
begin
   if n > 2 then
      F := F(n - 1) + G(n - 2)
   else
      F := n;
end;
 function G(n: integer): integer;
begin
   if n > 2 then
      G := G(n - 1) + F(n - 2)
   else
      G := n+1;
end;
```

B4. Ниже записан алгоритм на языке Паскаль. Получив на вход число x, этот алгоритм печатает два числа: L и M. Укажите наименьшее число x, при вводе которого алгоритм печатает сначала 5, а потом 7.

```
var x, L, M: integer;
 begin
    readln(x);
    L := 0;
    M := 0;
    while x>0 do
   begin
        M := M + 1;          if x mod 2 <> 0 then
          L := L + 1;
         x := x div 2;
   end;
   writeln(L);
```

```
  writeln(M);
end.
```

Оценочный лист к типовому заданию В

B1	B2	B3	B4
77	-3	13	79

С1. Составьте оптимальную по времени и использованию памяти программу, проверяющую можно ли представить число n в виде суммы двух простых слагаемых. Если такое возможно, то выводящую эти числа на экран.

Оценочный лист к типовому заданию С (модельный ответ)

С1. Пример решения программы:

```
Program C1;
var i,n:integer; l:boolean;
function prost (a:integer):boolean;
var j:integer;b:boolean;
begin
b:=true;j:=2;
while (j<=round(sqrt(a))) and b do begin
if a mod j=0 then b:=false;
inc(j);
prost:=b;
end;
end;
begin
writeln ('введите  N');
readln(n);
l:=true;
for i:=2 to n div 2 do
if prost(i) and prost(n-i) then begin writeln (i,' ',n-i); l:=false; end;
if l then writeln ('Нельзя представить')
end.
```

Критерии правильного ответа к заданию С1

1. Программа выдает верный ответ как в случае, если число можно представить в виде суммы двух простых слагаемых, так и в противном случае.
2. Ответ «нельзя представить в виде двух простых слагаемых» выводится за циклом (то есть один раз).
3. Проверка числа на «простоту» осуществляется в подпрограмме.
4. Проверка на «простоту» прекращается после нахождения первого делителя числа отличного от двойки.

Методические материалы, определяющие процедуру и критерии оценивания сформированности компетенций при проведении промежуточной аттестации

Код контролируемой компетенции (индикаторы)	Наименование оценочного средства	Максимальное количество баллов	Всего баллов	Уровень освоения компетенции (в баллах)		
				Пороговый (56-70%)	Продвинутый (71-85%)	Высокий (86-100%)
УК-1.1	Задание А 1-10	10	10	5-6	7-8	9-10
УК-1.2	Задание А 11-20	10	10	5-6	7-8	9-10
УК-1.3	Задание В 1-2	12	12	5-6	7-8	9-10

Код контролируемой компетенции (индикаторы)	Наименование оценочного средства	Максимальное количество баллов	Всего баллов	Уровень освоения компетенции (в баллах)		
				Пороговый (56-70%)	Продвинутый (71-85%)	Высокий (86-100%)
УК-1.4	Задание В 3-4	12	12	5-6	7-8	9-10
УК-1.5	Задание С	22	22	9-11	12-13	14-16

Полученное число баллов (27-66) выставляется в графу «Промежуточная аттестация» балльно-рейтинговой карты дисциплины.

Комплект оценочных средств для проведения промежуточной аттестации 2 семестр

Раздел «Структуры данных»

Проверяемая компетенция:
Универсальная компетенция УК-1.
Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач.

Проверяемый индикатор достижения компетенции:
УК-1.1: анализирует задачу, выделяя этапы ее решения, действия по решению задачи.

Проверяемые результаты обучения:
Знает: этапы решения задачи на компьютере.
Умеет: анализировать задачу, выделяя её базовые составляющие; осуществлять декомпозицию задачи

Проверяемый индикатор достижения компетенции:
УК-1.2 Находит, критически анализирует и выбирает информацию, необходимую для решения

Проверяемые результаты обучения:
Знает: способы формализации алгоритмов на языках программирования высокого уровня.
Умеет: осуществлять постановку задачи; анализировать условие и определять оптимальный метод решения поставленной задачи.

Проверяемый индикатор достижения компетенции:
УК-1.3. Рассматривает различные варианты решения задачи, оценивает их преимущества и риски

Проверяемые результаты обучения:
Знает: методы процедурного, объектно-ориентированного, функционального и визуального программирования; основные конструкции языков программирования; основные типы данных и операторы; приемы оптимизации алгоритмов по памяти и времени.
Умеет: строить математическую модель; составлять алгоритм решения задачи; записывать разрабатываемые алгоритмы на языках программирования высокого уровня.

Проверяемый индикатор достижения компетенции:
УК-1.4. Грамотно, логично, аргументированно формирует собственные суждения и оценки; отличает факты от мнений, интерпретаций, оценок в рассуждениях других участников деятельности.

Знает: основные виды ошибок, возникающих при решении задачи.
Умеет: комментировать синтаксические и семантические ошибки, возникающие при некорректном выполнении программы; отлаживать и тестировать задачи; составлять систему тестов для автоматизированной проверки корректности программы.

Проверяемый индикатор достижения компетенции:

УК-1.5. Определяет и оценивает практические последствия возможных вариантов решения задачи

Проверяемые результаты обучения:
Умеет: выполнять оценку сложности алгоритмов, проводить анализ и оценивание полученных результатов.

Тип (форма) задания: тестовые задания с выбором одного или нескольких вариантов ответа (A), практические задания (B, C).

Содержание задания:

A1. Структура данных – это
a) список данных, в котором порядок элементов списка задан посредством указателей, включенных в их запись;
b) множество элементов данных, объединенных и упорядоченных определенным образом;
c) представление логической организации данных в виде множества типов записей данных и связей между ними
d) список данных, в котором порядок элементов никак не обозначен.

A2. Укажите наиболее полный перечень способов записи алгоритмов
a) словесный;
b) словесный, графический, псевдокод, программный;
c) графический, программный;
d) словесный, программный.

A3. Суть такого свойства алгоритма, как дискретность заключается в том, что
a) алгоритм всегда состоит из последовательности дискретных шагов;
b) для записи алгоритма используются команды, которые входят в систему команд исполнителя;
c) алгоритм обеспечивает решение не одной конкретной задачи, а некоторого класса задач;
d) при точном исполнении всех команд алгоритма процесс должен прекратиться за конечное число шагов и привести к определенному результату.

A4. Алгоритм называется линейным, если
a) он составлен так, что его выполнение предполагает многократное повторение одних и тех же действий;
b) последовательность выполнения его команд зависит от истинности тех или иных условий;
c) его команды выполняются в порядке их естественного следования друг за другом независимо от каких-либо условий;
d) он представим в табличной форме.

A5. Системы программирования
a) обеспечивают непосредственное решение пользовательских задач;
b) позволяют создавать новые программы на языках программирования;
c) обеспечивают работу всех аппаратных устройств компьютера и доступ пользователя к ним;
d) представляют собой совокупность программ, используемых для различных операций с документами.

A6. Переменная в программировании наиболее полно характеризуется
a) именем;
b) именем, значением и типом;
c) именем и типом;
d) значением.

A7. Формальные параметры процедуры:
a) описываются в ее заголовке;
b) перечисляются при вызове процедуры;
c) указываются при описании данных в программе;
d) указываются при описании внутренних переменных процедуры.

A8. В какую из перечисленных ниже структур можно объединять данные различного типа
a) запись;
b) файл;
c) массив;
d) множество?

A9. Запись – это
a) именованный набор с фиксированным количеством однотипных данных;
b) совокупность разнородных данных, описываемых и обрабатываемых как единое целое;
c) ограниченная апострофами последовательность любых символов;
d) множество элементов данных, объединенных и упорядоченных определенным образом.

А10. Какой язык относится к процедурным языкам программирования
a) Pascal;
b) Visual Basic;
c) Лого;
d) Java?

А11. Результатом вычисления функции Copy(‘информатика’,3,5) будет слово
a) фор;
b) рма;
c) инфор;
d) форма.

А12. Определите, какому условию задачи соответствует данная программа:
```
Var x:string; i,j,s,t:integer;
Begin   readln(x); s:=0;   for i:=1 to length(x) do
begin
t:=0;
for j:=1 to length(x) do
If x[i]=x[j] then t:=t+1;
if t>1 then s:=s+1/t;     end;   writeln(t); End.
```
a) составьте программу, подсчитывающую количество различных символов в строке X;
b) составьте программу, подсчитывающую количество неповторяющихся символов строки X;
c) составьте программу, подсчитывающую количество повторяющихся символов в строке X;
d) составьте программу, подсчитывающую количество символов в строке X.

А13. При наборе программы вычисления суммы отрицательных элементов массива вместо оператора s:=s+a[k] ошибочно был записан оператор s:=s+1. Каким оказался ответ после исполнения неверной программы, если в качестве элементов массива были введены числа: –1, 3, –2, 4, –5, 6, –7, 8?
```
Var a: array[1..8] of Integer; s, k: Integer;
Begin
  For k:=1 to 8 Do  Readln(a[k]);    s:=0;
  For k:=1 to 8 Do I     f a[k]<0 Then s:=s+a[k]; Writeln(s) End
```
a) –15;
b) –3;
c) 4;
d) 10.

А14. Чему будет равно K после исполнения фрагмента программы:
```
K:=1;
While (A[K]<>X) AND (K<=10) Do K:=K+1;
```
если в качестве элементов массива будут введены числа 2, 3, 5, 7, 9, 12, 0, 7, 6, 7, а X=7
a) 14;
b) 1;
c) 10;
d) 8?

А15. Определите, какая ошибка была допущена в программе, предназначенной для подсчета неотрицательных элементов двумерного массива.
```
Var  A: array [1..10, 1..10] of real; N,I,J,t:integer; begin readln(N);
  for I:=1 to N do for J:=1 to N do readln(A[I,J]); t:=0;
  for i:=1 to N do for j:=1 to N do     if a[I,J]>0 then t:=t+1;   writeln(t); end.
```
a) программа подсчитывает количество отрицательных элементов;
b) программа вычисляет сумму элементов;
c) не учитываются элементы со значением «0»;
d) программа не выводит результат.

А16. Физическим именем файла в процедурном языке программирования называют:
a) имя переменной, использующейся в программе при осуществлении операций над файлом;
b) имя программы обработки файла;
c) имя, под которым программа обработки файла хранится на диске;
d) имя файла, под которым он записан на внешнем устройстве.

А17. Открывает существующий файл процедура

a) reset;
b) rewrite;
c) close;
d) rename.

A18. Определите, какому условию задачи соответствует представленная программа.

```
var w:text; s:integer; a:char;
begin
  s:=0; assign(w,'a.txt'); reset(w);     while not eof(w) do
    begin        if eoln(w) then s:=s+1;         read(w,a); write(a)
    end;    close(w); write(s) end.
```

a) вывести текстовый файл на экран и подсчитать количество строк в данном текстовом файле;
b) вывести текстовый файл на экран и подсчитать количество символов в данном текстовом файле;
c) создать текстовый файл и подсчитать количество символов в нем;
d) подсчитать количество символов в файле.

A19. Каким оказался ответ после исполнения программы:

```
const N=10;
type T1=1..N;
var A,B,C: set of T1; j: integer;
begin
A:=[1,2,3,5,6]; B:=[1,4,6]; C:=A*B;
for j:=1 to N do
     if j in C   then write(j,'  '); end.
```

a) 6;
b) 1 2 3 4 5 6;
c) 2 3 5;
d) 4?

A20. В процедурном языке программирования основное различие между процедурами и функциями заключается в том, что:

a) в процедуре допускается описание локальных переменных, а в функции – нет;
b) в программе обращение к процедуре может осуществляться многократно, тогда как к функции только один раз;
c) в процедуре допускается использование глобальных переменных, а в функции – нет;
d) в результате работы процедуры можно получить любое количество данных, а функции – только одно.

Оценочный лист к типовому заданию А

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
a	b	a	c	b	b	a	a	b	a
A11	A12	A13	A14	A15	A16	A17	A18	A19	A20
d	c	c	d	c	d	a	a	a	d

Содержание задания:

B1. В программе используется одномерный целочисленный массив A с индексами от 0 до 9. Значения элементов равны 3, 0, 4, 6, 5, 1, 8, 2, 9, 7 соответственно, т.е. A[0] = 3, A[1] = 0 и т.д. Определите значение переменной c после выполнения следующего фрагмента этой программы

```
c := 0
нц для i от 1 до 9
если A[i-1] > A[i] то
 c := c + 1
 t := A[i]
 A[i] := A[i-1]
 A[i-1] := t
все
кц
```

B2. Запишите подряд все числа, которые будут напечатаны на экране при выполнении вызова F(9). Числа должны быть записаны в том же порядке, в котором они выводятся на экран.

```
procedure F(n: integer);
begin
 if n > 0 then
 begin
 write(n);
```

```
 F(n - 3);
 F(n div 3)
 end
end;
```

В3 Какие ошибки допущены в следующем описании массива:
Var a: array[1…5.5] of char;

Оценочный лист к типовому заданию В

В1	В2	В3
5	9 6 3 1 2 3 1	Диапазон задается двумя точками, а не тремя, индексом не может быть дробное число

С1. Опишите любой из известных вам алгоритмов сортировки массива целых чисел по убыванию.

С2. Дан целочисленный массив из 30 элементов. Элементы массива могут принимать целые значения от 0 до 10 000 включительно. Напишите программу на процедурном языке программирования, которая находит количество элементов массива, больших 100 и при этом кратных 5, а затем заменяет каждый такой элемент на число, равное найденному количеству. Гарантируется, что, хотя бы один такой элемент в массиве есть. В качестве результата необходимо вывести измененный массив, каждый элемент массива выводится с новой строчки. Например, для массива из шести элементов: 4 115 7 195 25 106 программа должна вывести числа 4 2 7 2 25 106

С3. Расскажите основные идеи метода быстрой сортировки (изучается в углубленном курсе информатики), опишите алгоритм.

Оценочный лист к типовому заданию С

С1. Допускается следующий вариант ответа:

1. Для исходного массива выбрать минимальный элемент.
2. Поменять его местами с последним элементом (после этого самый большой элемент будет стоять на своем месте).
3. Повторить п.п. 1-2 с оставшимися n-1 элементами, то есть рассмотреть часть массива, начиная с первого элемента до предпоследнего, найти в нем минимальный элемент и поменять его местами с предпоследним (n-1)- м элементом массива, затем с оставшиеся (n-2)-мя элементами и так далее, пока не останется один элемент, уже стоящий на своем месте.

Представлен алгоритм сортировки методом простого выбора. Допускается также алгоритм сортировки методом пузырька и др.

С2. Допускается следующий вариант ответа:

```
Const N = 30;
var
a: array [1..N] of longint;
i, j, k: longint;
begin
 for i := 1 to N do
   readln(a[i]);
k := 0;
for i := 1 to N do
  if (a[i] > 100) and (a[i] mod 5 = 0) then
     k:=k+1;
for i := 1 to N do begin
    if (a[i] > 100) and (a[i] mod 5 = 0) then
       a[i] := k;
 writeln(a[i])
end.
```

С3. Быстрая сортировка является существенно улучшенным вариантом алгоритма сортировки с помощью прямого обмена. Принципиальное отличие состоит в том, что в первую очередь производятся перестановки на наибольшем возможном расстоянии и после каждого прохода элементы делятся на две независимые группы.

Алгоритм состоит из трёх шагов:

1. Выбрать опорный элемент из массива.
2. Разбиение: перераспределение элементов в массиве таким образом, что элементы меньше опорного помещаются перед ним, а больше или равные после.

3. Рекурсивно применить первые два шага к двум подмассивам слева и справа от опорного элемента. Рекурсия не применяется к массиву, в котором только один элемент или отсутствуют элементы.

Методические материалы, определяющие процедуру и критерии оценивания сформированности компетенций при проведении промежуточной аттестации

Код контролируемой компетенции (индикаторы)	Наименование оценочного средства	Максимальное количество баллов	Всего баллов	Уровень освоения компетенции (в баллах)		
				Пороговый (56-70%)	Продвинутый (71-85%)	Высокий (86-100%)
УК-1.1	Задание А 1-10	10	10	5-6	7-8	9-10
УК-1.2	Задание А 11-20	10	10	5-6	7-8	9-10
УК-1.3	Задание В 1	5	5	2-3	4	5
УК-1.4	Задание В 2-3	20	20	10-14	15-16	17-20
УК-1.5	Задание С	15	15	8-10	11-13	14-15

Полученное число баллов (36-60) выставляется в графу «Промежуточная аттестация» балльно-рейтинговой карты дисциплины.