МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М. В. Ломоносова

Факультет
вычислительной математики
и кибернетики

М.А. КАЗАЧУК, И.В. МАШЕЧКИН, И.С. ПОПОВ

Тестовые задания по курсу «ОПЕРАЦИОННЫЕ СИСТЕМЫ»

Учебно-методическое пособие

МАКС ПРЕСС

МОСКВА – 2022

УДК 681.3.06(075.8)
ББК 32.973.1я73
К14

https://elibrary.ru/pflcla

Печатается по решению Редакционно-издательского Совета факультета вычислительной математики и кибернетики МГУ имени М. В. Ломоносова

Рецензенты:
И.Г. Головин – к.ф.-м.н., доцент, факультет Вычислительной математики и кибернетики МГУ имени М.В. Ломоносова;
Е.А. Кузьменкова – к.ф.-м.н., доцент, факультет Вычислительной математики и кибернетики МГУ имени М.В. Ломоносова

Казачук М.А., Машечкин И.В., Попов И.С.

К14 **Тестовые задания по курсу «Операционные системы» :** учебно-методическое пособие / М.А. Казачук, И.В. Машечкин, И.С. Попов. – Москва : МАКС Пресс, 2022. – 164 с.
ISBN 978-5-317-06863-9

Пособие подготовлено авторами для поддержки курса «Операционные системы», читаемого в третьем семестре на факультете ВМК МГУ, и предназначено для проверки знаний студентов и подготовки к экзамену. В данном пособии предложены комбинации теоретических вопросов и задач на программирование на языке Си по программе лекционного курса.

Пособие разработано на основе базового набора тестовых заданий и их модификаций, подготовленных преподавателями факультета ВМК МГУ: Волковой И.А., Вылитком А.А., Глазковой В.В., Гомзиным А.Г., Жуковым К.А., Казачук М.А., Корныхиным Е.В., Кузиной Л.Н., Санжаровым В.В., Тюляевой В.В., Черновым А.В.

Ключевые слова: операционные системы, процессы, взаимодействие процессов, планирование выполнения процессов.

УДК 681.3.06(075.8)
ББК 32.973.1я73

ISBN 978-5-317-06863-9

Содержание

Введение

Пособие подготовлено авторами для поддержки курса «Операционные системы», читаемого в третьем семестре на факультете ВМК МГУ, и предназначено для проверки знаний студентов и подготовки к экзамену. В данном пособии предложены комбинации теоретических вопросов и задач на программирование на языке Си по программе лекционного курса.

Пособие разработано на основе базового набора тестовых заданий и их модификаций, подготовленных преподавателями факультета ВМК МГУ: Волковой И.А., Вылитком А.А., Глазковой В.В., Гомзиным А.Г., Жуковым К.А., Казачук М.А., Корныхиным Е.В., Кузиной Л.Н., Санжаровым В.В., Тюляевой В.В., Черновым А.В.

Задача 1

№	Условие	Ответ
1	В оперативном запоминающем устройстве 16-ти разрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове восьмеричного числа $(17735)_8$.	$(17735)_8$. Посчитаем число двоичных единиц в каждой восьмеричной цифре: 1 + 3 + 3 + 2 + 2 = 11, число нечетное. Тогда **тег контроля четности 1.**
2	В оперативном запоминающем устройстве 16-ти разрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове восьмеричного числа $(35735)_8$.	$(35735)_8$. Посчитаем число двоичных единиц в каждой восьмеричной цифре: 2 + 2 + 3 + 2 + 2 = 11, число нечетное. Тогда **бит паритета равен 1.** Структура ячейки памяти: 16 бит данных (0011101111011101) + 1 бит паритета (1), который вычисляется как сумма по модулю 2 (XOR) всех битов данных.
3	В оперативном запоминающем устройстве 16-ти разрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове восьмеричного числа $(26775)_8$.	$(26775)_8$. Посчитаем число двоичных единиц в каждой восьмеричной цифре: 1 + 2 + 3 + 3 + 2 = 11, число нечетное. Тогда **тег контроля четности 1.**
4	В оперативном запоминающем устройстве 16-ти разрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для	$(1033231)_4$. Посчитаем число двоичных единиц в каждой четверичной цифре: 1 + 0 + 2 + 2 + 1 + 2 + 1 = 9, число нечетное. Тогда **тег контроля четности 1.**

№	Условие	Ответ
	случая хранения в машинном слове четверичного числа $(1033231)_4$.	
5	В оперативном запоминающем устройстве 32-ух разрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове восьмеричного числа $(3560271)_8$.	$(3560271)_8$. Посчитаем число двоичных единиц в каждой восьмеричной цифре: 2 + 2 + 2 + 0 + 1 + 3 + 1 = 11, число нечетное. Тогда **тег контроля четности 1.**
6	В оперативном запоминающем устройстве 32-хразрядного компьютера используется контроль целостности данных по нечетности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове 16-ичного числа $(FF001077)_{16}$.	$(FF001077)_{16}$. Посчитаем число двоичных единиц в каждой ненулевой 16-ичной цифре: 4+4+1+3+3 = 15, число нечетное. Тогда **тег контроля четности 1.**
7	В оперативном запоминающем устройстве 32-хразрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове шестнадцатеричного числа $(FF001033)_{16}$.	$(FF001033)_{16}$. Посчитаем число двоичных единиц в каждой ненулевой 16-ичной цифре: 4+4+1+2+2 = 13, число нечетное. Тогда **тег контроля четности 1.**
8	В оперативном запоминающем устройстве 8-ми разрядного встроенного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове шестнадцатеричного числа $(A1)_{16}$.	$(A1)_{16}$. Посчитаем число двоичных единиц в каждой шестнадцатеричной цифре: 2 + 1 =3, число нечетное. Тогда **тег контроля четности 1.**

№	Условие	Ответ
9	В оперативном запоминающем устройстве 32-ти разрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове восьмеричного числа $(1313131313)_8$.	$(1313131313)_8$. Посчитаем число двоичных единиц в каждой восьмеричной цифре: 15, число нечетное. Тогда **тег контроля четности 1.**
10	В оперативном запоминающем устройстве 16-ти разрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове восьмеричного числа $(27435)_8$.	число дв единиц 1+3+1+2+2=9 тег контроля четности 1
11	В оперативном запоминающем устройстве 16-ти разрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове восьмеричного числа $(32117)_8$.	число дв единиц 2+1+1+1+3=8 тег контроля четности 0
12	В оперативном запоминающем устройстве 16-ти разрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове восьмеричного числа $(27463)_8$.	число дв единиц 1+3+1+2+2=9 тег контроля четности 1

№	Условие	Ответ
13	В оперативном запоминающем устройстве 16-ти разрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове восьмеричного числа $(37432)_8$.	$(37432)_8$. Посчитаем число двоичных единиц в каждой восьмеричной цифре: 2 + 3 + 1 + 2 + 1 = 9, число нечетное. Тогда **тег контроля четности 1.** 0 011 111 100 011 010 + 1
14	В оперативном запоминающем устройстве 16-ти разрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове восьмеричного числа $(21345)_8$.	$(21345)_8$. Посчитаем число двоичных единиц в каждой восьмеричной цифре: 1 + 1 + 2 + 1 + 2 = 7, число нечетное. Тогда **тег контроля четности 1.** 0 010 001 011 100 101 + 1
15	В оперативном запоминающем устройстве 16-ти разрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове восьмеричного числа $(12467)_8$.	$(12467)_8$. Посчитаем число двоичных единиц в каждой восьмеричной цифре: 1 + 1 + 1 + 2 + 3 = 8, число четное. Тогда **тег контроля четности 0.** 0 001 010 100 110 111 + 0
16	В оперативном запоминающем устройстве 16-ти разрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове восьмеричного числа $(31746)_8$.	$(31746)_8$. Посчитаем число двоичных единиц в каждой восьмеричной цифре: 2 + 1 + 3 + 1 + 2 = 9, число нечетное. Тогда **тег контроля четности 1.** Машинное слово: **1** 0 011 001 111 100 110

№	Условие	Ответ
17	В оперативном запоминающем устройстве 14-ти разрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове восьмеричного числа 216_8.	00000010001110, бит четности: 0

Задача 2

№	Условие	Ответ
1	Пусть дано восьмеричное число $(173357)_8$, являющееся адресом оперативной памяти, расслоенной по 16 банкам. Банку с каким номером принадлежит заданный адрес?	$(173357)_8$. В схеме расслоения памяти последовательные адреса размещаются в последовательных банках памяти. При 16 банках памяти за номер банка будут отвечать младшие 4 бита адреса. Они равны 1111, то есть 15. Ответ: **банк памяти 15**. (нумерация банков памяти с 0)
2	Пусть дано восьмеричное число $(173305)_8$, являющееся адресом оперативной памяти, расслоенной по 16 банкам. Банку с каким номером принадлежит заданный адрес?	$(173305)_8$. В схеме расслоения памяти последовательные адреса размещаются в последовательных банках памяти. При 16 банках памяти за номер банка будут отвечать младшие 4 бита адреса. Они равны 0101, то есть 5. Ответ: **банк памяти 5**.

№	Условие	Ответ
3	Пусть дано восьмеричное число $(173367)_8$, являющееся адресом оперативной памяти, расслоенной по 32 банкам. Банку с каким номером принадлежит заданный адрес?	$(173367)_8$. В схеме расслоения памяти последовательные адреса размещаются в последовательных банках памяти. При 32 банках памяти за номер банка будут отвечать младшие 5 битов адреса. Они равны 10111, то есть 23. Ответ: **банк памяти 23**. (нумерация банков памяти с 0)
4	Пусть дано восьмеричное число $(4321475)_8$, являющееся адресом оперативной памяти, расслоенной по 4 банкам. Банку с каким номером принадлежит заданный адрес?	$(4321475)_8$. В схеме расслоения памяти последовательные адреса размещаются в последовательных банках памяти. При 4 банках памяти за номер банка будут отвечать младшие 2 бита адреса. Они равны 01, то есть 1. Ответ: **банк памяти 1**. (нумерация банков памяти с 0)
5	Пусть дано четверичное число $(323112)_4$, являющееся адресом оперативной памяти, расслоенной по 8 банкам. Банку с каким номером принадлежит заданный адрес?	$(323112)_4$. В схеме расслоения памяти последовательные адреса размещаются в последовательных банках памяти. При 8 банках памяти за номер банка будут отвечать младшие 3 бита адреса. Они равны 110, то есть 6. Ответ: **банк памяти 6**. (нумерация банков памяти с 0)
6	Пусть дано 16-ичное число $(FAD1D31A)_{16}$, являющееся адресом оперативной памяти, расслоенной по 16 банкам. Банку с каким номером принадлежит заданный адрес?	$(FAD1D31A)_{16}$. В схеме расслоения памяти последовательные адреса размещаются в последовательных банках памяти. При 16 банках памяти за номер банка будет отвечать младшая 16-ичная цифра адреса. Она равны

№	Условие	Ответ
		$0A_{16}$, то есть 10. Ответ: **банк памяти 10**. (нумерация банков памяти с 0)
7	Пусть дано 16-ичное число $(FAD1D319)_{16}$, являющееся адресом оперативной памяти, расслоенной по 8 банкам. Банку с каким номером принадлежит заданный адрес?	$(FAD1D319)_{16}$. В схеме расслоения памяти последовательные адреса размещаются в последовательных банках памяти. При 8 банках памяти за номер банка будут отвечать младшие 3 бита адреса. Они равны 001, то есть 1. Ответ: **банк памяти 1**. (нумерация банков памяти с 0)
8	Пусть дано четверичное число $(123123)_4$, являющееся адресом оперативной памяти, расслоенной по 8 банкам. Банку с каким номером принадлежит заданный адрес?	$(123123)_4$. В схеме расслоения памяти последовательные адреса размещаются в последовательных банках памяти. При 8 банках памяти за номер банка будут отвечать младшие 3 бита адреса. Они равны 011_2, то есть 3. Ответ: **банк памяти 3**. (нумерация банков памяти с 0)
9	Пусть дано восьмеричное число $(125432)_8$, являющееся адресом оперативной памяти, расслоенной по 8 банкам. Банку с каким номером принадлежит заданный адрес?	$(125432)_8$. В схеме расслоения памяти последовательные адреса размещаются в последовательных банках памяти. При 8 банках памяти за номер банка будут отвечать младшие 3 бита адреса. Они равны 010, то есть 2. Ответ: **банк памяти 2**. (нумерация банков памяти с 0)
10	Пусть дано восьмеричное число $(213417)_8$, являющееся адресом оперативной памяти, расслоенной по 16 банкам. Банку с каким номером принадлежит заданный адрес?	младшие 4 бита: 1111 **банк памяти 15**

№	Условие	Ответ
11	Пусть дано восьмеричное число $(376154)_8$, являющееся адресом оперативной памяти, расслоенной по 16 банкам. Банку с каким номером принадлежит заданный адрес?	младшие 4 бита: 1100 **банк памяти 12**
12	Пусть дано восьмеричное число $(124572)_8$, являющееся адресом оперативной памяти, расслоенной по 16 банкам. Банку с каким номером принадлежит заданный адрес?	младшие 4 бита: 1010 **банк памяти 10**
13	Пусть дано восьмеричное число $(123456)_8$, являющееся адресом оперативной памяти, расслоенной по 16 банкам. Банку с каким номером принадлежит заданный адрес?	$(123456)_8$. В схеме расслоения памяти последовательные адреса размещаются в последовательных банках памяти. При 16 банках памяти за номер банка будут отвечать младшие 4 бита адреса. Они равны 1110, то есть 14. Ответ: **банк памяти 14**. (нумерация банков памяти с 0)
14	Пусть дано восьмеричное число $(234432)_8$, являющееся адресом оперативной памяти, расслоенной по 16 банкам. Банку с каким номером принадлежит заданный адрес?	$(234432)_8$. В схеме расслоения памяти последовательные адреса размещаются в последовательных банках памяти. При 16 банках памяти за номер банка будут отвечать младшие 4 бита адреса. Они равны 1010, то есть 10. Ответ: **банк памяти 10**. (нумерация банков памяти с 0)
15	Пусть дано восьмеричное число $(143341)_8$, являющееся адресом оперативной памяти, расслоенной по 16 банкам. Банку с каким номером принадлежит заданный адрес?	$(143341)_8$. В схеме расслоения памяти последовательные адреса размещаются в последовательных банках памяти. При 16 банках памяти за номер банка будут отвечать

№	Условие	Ответ
		младшие 4 бита адреса. Они равны 0001, то есть 1. Ответ: **банк памяти 1**. (нумерация банков памяти с 0)
16	Пусть дано шестнадцатеричное число $(AABB)_{16}$, являющееся адресом оперативной памяти, расслоенной по 16 банкам. Банку с каким номером принадлежит заданный адрес?	$(AABB)_{16}$ В схеме расслоения памяти последовательные адреса размещаются в последовательных банках памяти. При 16 банках памяти за номер банка будут отвечать младшие 4 бита адреса. Они равны 1011, то есть 11. Ответ: банк памяти 11. (нумерация банков памяти с 0)
17	Пусть дано шестнадцатеричное число $(E57A)_{16}$, являющееся адресом оперативной памяти, расслоенной по 8 банкам. Банку с каким номером принадлежит заданный адрес?	$(E57A)_{16}$. В схеме расслоения памяти последовательные адреса размещаются в последовательных банках памяти. При 8 банках памяти за номер банка будут отвечать младшие 3 бита адреса. Они равны 010, то есть 2. Ответ: **банк памяти 2**. (нумерация банков памяти с 0)
18	Пусть дано восьмеричное число 170120_8, являющееся адресом оперативной памяти, расслоенной по 16 банкам. Банку с каким номером принадлежит заданный адрес?	0 (банки нумеруются от 0)
19	На некотором компьютере используется расслоение оперативной памяти по 16 банкам. Адресация «плоская». Ячейка имеет восьмеричный адрес 3762315453_8 В каком банке она находится? (банки нумеруются с нуля).	В 11-ом (младшие 4 разряда: 1011)

Задача 3

№	Условие	Ответ
1	Дан 32-х разрядный IP адрес, имеющий в восьмеричном представлении вид: $(23171171543)_8$. Определить: к какому классу относится данный IP адрес; номер сети (в восьмеричном представлении), к которой относится IP адрес.	$(23171171543)_8$. **10**0111001111001001**111001101100011**$_2$ Старшие биты адреса: 10, это сеть класса **B**. Номер сети – следующие 14 бит, номер хоста – оставшиеся 16 бит. Тогда номер сети: 01100111100100_2 = **14744**$_8$.
2	Дан 32-х разрядный IP адрес, имеющий в восьмеричном представлении вид: $(27151171543)_8$. Определить: к какому классу относится данный IP адрес; номер сети (в восьмеричном представлении), к которой относится IP адрес.	$(27151171543)_8$. **10**1110011010010**111001101100011**$_2$ Старшие биты адреса: 10, это сеть класса **B**. Номер сети – следующие 14 бит, номер хоста – оставшиеся 16 бит. Тогда номер сети: 11100110100100_2 = **34644**$_8$.
3	Дан 32-х разрядный IP адрес, имеющий в восьмеричном представлении вид: $(33171171543)_8$. Определить: к какому классу относится данный IP адрес; номер сети (в восьмеричном представлении), к которой относится IP адрес.	$(33171171543)_8$. **110**11001111001001111001**11100011**$_2$ Старшие биты адреса: 110, это сеть класса **C**. Номер сети – следующие 21 бит, номер хоста – оставшиеся 8 бит. Тогда номер сети: $11001111001001111001 1_2$ = **6362363**$_8$.
4	Дан 32-х разрядный IP адрес, имеющий в 16-ичном представлении вид: $(DF00BE20)_{16}$. Определить: к какому классу относится данный IP адрес; номер сети (в 16-	$(DF00BE20)_{16}$. **110**11111000000001011111**00100000**$_2$

№	Условие	Ответ
	ичном представлении), и десятичный номер хоста в сети, к которой относится IP адрес.	Старшие биты адреса: 110, это сеть класса **C**. Номер сети – следующие 21 бит, номер хоста – оставшиеся 8 бит. Тогда номер сети: $11111000000000101111110_2$ = $1F00BE_{16}$, а номер хота – 32 (20_{16})
5	Дан 32-х разрядный IP адрес, имеющий в восьмеричном представлении вид: $(22011171543)_8$. Определить: к какому классу относится данный IP адрес; номер сети (в десятичном представлении), к которой относится IP адрес.	$(22011171543)_8$. **10** 010 000 001 001 00**1 111 001 101 100 011**$_2$ Старшие биты адреса: 10, это сеть класса **B**. Номер сети – следующие 14 бит, номер хоста – оставшиеся 16 бит. Тогда номер сети: $01000000100100_2 = 2^{12} + 32 + 4$ $=4096+36=$**4132_{10}**.
6	Дан 32-х разрядный IP адрес, имеющий в восьмеричном представлении вид: $(17171171543)_8$. Определить: к какому классу относится данный IP адрес; номер сети (в восьмеричном представлении), к которой относится IP адрес.	$(17171171543)_8$. **0**1111001**11100100111100110110011**$_2$ Старшие биты адреса: 0, это сеть класса **A**. Номер сети – следующие 7 бит, номер хоста – оставшиеся 24 бита. Тогда номер сети: 1111001_2 = **171_8**.
7	Дан 32-х разрядный IP адрес, имеющий в восьмеричном представлении вид: $(17636744535)_8$. Определить: к какому классу относится данный IP адрес; номер сети (в восьмеричном представлении), к которой относится IP адрес.	$(17636744535)_8$ = **0**1111110**01111011 11001001 01011101**$_2$ Старший бит адреса: 0, это сеть класса **A**. Номер сети – следующие 7 бит, номер хоста – оставшиеся 24 бита. Тогда номер сети: 1111110_2 = **$(176)_8$**

№	Условие	Ответ
8	Дан 32-х разрядный IP адрес, имеющий в восьмеричном представлении вид: $(23164742575)_8$. Определить: к какому классу относится данный IP адрес; номер сети (в восьмеричном представлении), к которой относится IP адрес.	$(23164742575)_8$ = **10**011001 11010011 **11000101 01111101**$_2$ Старшие биты адреса: 10, это сеть класса **B**. Номер сети – следующие 14 бит, номер хоста – оставшиеся 16 бит. Тогда номер сети: 01100111010011_2 = **(14723)**$_8$.
9	Дан 32-х разрядный IP адрес, имеющий в восьмеричном представлении вид: $(32310543247)_8$. Определить: к какому классу относится данный IP адрес; номер сети (в восьмеричном представлении), к которой относится IP адрес.	$(32310543247)_8$ = **110**10011 00100010 11000110 **10100111**$_2$ Старшие биты адреса: 110, это сеть класса **C**. Номер сети – следующие 21 бит, номер хоста – оставшиеся 8 бит. Тогда номер сети: $(10011001000101100011 0)_2$ = $(4621306)_8$
10	Дан 32-х разрядный IP адрес, имеющий в восьмеричном представлении вид: $(33221177543)_8$. Определить: к какому классу относится данный IP адрес; номер сети (в восьмеричном представлении), к которой относится IP адрес.	$(33221177543)_8$. **(11 011 010 010 001 001 111 111 101 100 011)**$_2$ Старшие биты адреса: 110, это сеть класса **C**. Номер сети – следующие 21 бит, номер хоста – оставшиеся 8 бит. Тогда номер сети: **(110 100 100 010 011 111 111)**$_2$ = **6442377**$_8$.
11	Дан 32-х разрядный IP адрес, имеющий в восьмеричном представлении вид: $(21176543211)_8$. Определить: к какому классу относится данный IP адрес; номер сети (в восьмеричном представлении), к которой относится IP адрес.	$(21176543211)_8$. **(10 001 001 111 110 101 100 011 010 001 001)**$_2$ Старшие биты адреса: 10, это сеть класса **B**. Номер сети – следующие 14 бит, номер хоста –

№	Условие	Ответ
		оставшиеся 16 бит. Тогда номер сети: (**00 100 111 111 010**)$_2$ = **04772**$_8$.
12	Дан 32-х разрядный IP адрес, имеющий в восьмеричном представлении вид: (13206117253)$_8$. Определить: к какому классу относится данный IP адрес; номер сети (в восьмеричном представлении), к которой относится IP адрес.	(13206117253)$_8$. 01011010000110001001111010101011$_2$ Старшие бит адреса: 0, это сеть класса **A**. Номер сети – следующие 7 бит, номер хоста – оставшиеся 24 бит. Тогда номер сети: 1011010$_2$ = **132**$_8$.
13	Дан 32-х разрядный IP адрес, имеющий в восьмеричном представлении вид: 15502410544$_8$. Определить: к какому классу относится данный IP адрес; номер сети (в восьмеричном представлении), к которой относится IP адрес.	01 101 101 000 010 100 001 000 101 100 100 Сеть класса A. Номер сети: 155$_8$.

Задача 4

№	Условие	Ответ

1	Пусть процесс с PID **A** породил два сыновьих процесса с PID-ами **B** и **C**:
	int main(int argc, char **argv) //PID = A
	{
	if (fork() == 0){ //PID = B
	printf ("%d %d\n", getppid(), getpid());
	exit(0);
		A B
		C
		либо
		1 B
		C
		либо
		C

№	Условие	Ответ

	}
	if (fork() == 0){ //PID = C
	printf ("%d\n", getpid());
	exit(0);
	}
	return 0;
	}
	Считаем, что printf работает атомарно и обращения ко всем системным вызовам успешно отрабатывают.
	Перечислить **все** возможные комбинации значений, которые могут быть выведены на стандартное устройство вывода в результате выполнения данной программы.
		A B
		либо
		C
		1 B

2	Пусть процесс с PID **D** породил два сыновьих процесса с PID-ами **C** и **A**:
	int main(int argc, char **argv) //PID = D
	{
	if (fork() == 0){ //PID = C
	printf ("%d %d\n", getppid(), getpid());
	exit(0);
	}
	if (fork() == 0){ //PID = A
	printf ("%d\n", getpid());
	exit(0);
		D C
		A
		либо
		1 C
		A
		либо
		A
		D C
		либо
		A
		1 C

№	Условие	Ответ
	} return 0; } Считаем, что printf работает атомарно и обращения ко всем системным вызовам успешно отрабатывают. Перечислить **все** возможные комбинации значений, которые могут быть выведены на стандартное устройство вывода в результате выполнения данной программы.	

3	Пусть процесс с PID **A** породил сыновий процесс с PID **B**:
	int main(int argc, char **argv) //PID = A
	{
	int n = 42;
	if (fork() == 0){ //PID = B
	printf ("%d %d %d\n", n, getppid(), getpid());
	n = 8;
	exit(0);
	}
	n = 10;
	printf ("%d %d\n", n, getpid());
	return 0;
	}
		42 A B
		10 A
		либо
		10 A
		42 A B
		либо
		10 A
		42 1 B

№	Условие	Ответ
	Считаем, что printf работает атомарно, без буферизации, и обращения ко всем системным вызовам успешно отрабатывают. Перечислить **все** возможные комбинации значений, которые могут быть выведены на стандартное устройство вывода в результате выполнения данной программы.	

4	Пусть процесс с PID 4123 породил два сыновних процесса с PID-ами 4124 и 4125:
	int main(int argc, char **argv) //PID = 4123
	{
	if (fork() == 0){ //PID = 4124
	printf ("%d \n", getpid());
	exit(0);
	}
	wait(NULL);
	if (fork() == 0){ //PID = 4125
	printf ("%d %d \n", getpid(), getppid());
	exit(0);
	}
	return 0;
	}
	Считаем, что printf работает атомарно и обращения ко всем системным вызовам успешно отрабатывают. Перечислить **все** возможные комбинации значений,
		4124
		4125 4123
		либо
		4124
		4125 1

№	Условие	Ответ
	которые могут быть выведены на стандартное устройство вывода в результате выполнения данной программы.	

5	Пусть процесс с PID **A** породил два сыновьих процесса с PID-ами **F** и **B**:
	int main(int argc, char **argv) //PID = A
	{
	if (fork() == 0){ //PID = F
	printf ("%d %d\n", getppid(), getpid());
	exit(0);
	}
	if (fork() == 0){ //PID = B
	printf ("%d\n", getpid());
	exit(0);
	}
	return 0;
	}
	Считаем, что printf работает атомарно и обращения ко всем системным вызовам успешно отрабатывают.
	Перечислить **все** возможные комбинации значений, которые могут быть выведены на стандартное устройство вывода в результате выполнения данной программы.
		A F
		B
		либо
		1 F
		B
		либо
		B
		A F
		либо
		B
		1 F

№	Условие	Ответ

6	Пусть процесс с PID **X** породил два сыновьих процесса с PID-ами **Y** и **Z**:
	int main(int argc, char **argv) //PID = X
	{
	int pid;
	pid = getpid();
	if (fork() == 0){ //PID = Y
	printf ("%d %d\n", pid, getpid());
	exit(0);
	}
	if (fork() == 0){ //PID = Z
	printf ("%d\n", getppid());
	exit(0);
	}
	return 0;
	}
	Считаем, что printf работает атомарно и обращения ко всем системным вызовам успешно отрабатывают.
	Перечислить **все** возможные комбинации значений, которые могут быть выведены на стандартное устройство вывода в результате выполнения данной программы.
		X Y
		X
		либо
		X
		X Y
		либо
		X Y
		1
		либо
		1
		X Y

7	Пусть процесс с PID **P1** породил два сыновьих процесса с PID-ами **P2** и **P3**:	**P1 P2** **P3**

№	Условие	Ответ

	int main(int argc, char **argv) //PID = P1
	{
	if (fork() == 0){ //PID = P2
	printf ("%d %d\n", getppid(), getpid());
	exit(0);
	}
	if (fork() == 0){ //PID = P3
	printf ("%d\n", getpid());
	exit(0);
	}
	return 0;
	}
	Считаем, что printf работает атомарно и обращения ко всем системным вызовам успешно отрабатывают.
	Перечислить **все** возможные комбинации значений, которые могут быть выведены на стандартное устройство вывода в результате выполнения данной программы.
		либо
		1 P2
		P3
		либо
		P3
		P1 P2
		либо
		P3
		1 P2

8	Пусть процесс с PID **X** породил два сыновьих процесса с PID-ами **Y** и **Z**:
	int main(int argc, char **argv) //PID = X
	{
	if (fork() == 0){ //PID = Y
	printf ("%d %d\n", getppid(), getpid());
		X Y
		Z
		либо
		1 Y
		Z
		либо

№	Условие	Ответ

	exit(0);
	}
	if (fork() == 0){ //PID = Z
	printf ("%d\n", getpid());
	exit(0);
	}
	return 0;
	}
	Считаем, что printf работает атомарно и обращения ко всем системным вызовам успешно отрабатывают. Перечислить **все** возможные комбинации значений, которые могут быть выведены на стандартное устройство вывода в результате выполнения данной программы.
		Z
		X Y
		либо
		Z
		1 Y

9	Пусть процесс с PID **F** породил два сыновьих процесса с PID-ами **A** и **B**:
	int main(int argc, char **argv) //PID = F
	{
	if (fork() == 0){ //PID = A
	printf ("%d %d\n", getpid(), getppid());
	exit(0);
	}
	if (fork() == 0){ //PID = B
	printf ("%d\n", getpid());
		A F
		B
		либо
		A 1
		B
		либо
		B
		A F
		либо
		B

№	Условие	Ответ

	exit(0);
	}
	return 0;
	}
	Считаем, что printf работает атомарно и обращения ко всем системным вызовам успешно отрабатывают. Перечислить **все** возможные комбинации значений, которые могут быть выведены на стандартное устройство вывода в результате выполнения данной программы.
		A 1

Задача 5

№	Условие	Ответ
1	Пусть дана файловая система Unix System V и в ней утеряна информация суперблока. Предложить последовательность действий, позволяющую восстановить содержимое файлов данной файловой системы. Считаем, что до потери суперблока содержимое файловой системы было корректным. Размер суперблока, размер и структура индексного дескриптора известны.	Структура файловой системы версии System V: **{Суперблок} + {Область индексных дескрипторов} + {Блоки файлов}.** Размер области индексных дескрипторов хранится в суперблоке. При потере информации суперблока, данное значение теряется. Для восстановления содержимого файлов, необходимо определить границу между областью индексных дескрипторов и областью блоков файлов.

№	Условие	Ответ
		Идем итеративно от начала области индексных дескрипторов. Считываем очередной ID. Проверяем содержимое поля «ссылки на данный ID каталогов файловой системы». Если это поле равно нулю (это означает, что ID свободен) переходим к следующему ID. В противном случае последовательно просматриваем 13 элементов, описывающих адресацию блоков файла (до завершения): номера блоков с прямой адресацией (10 шт.), номера блоков, организованных с косвенной адресацией 1, 2 и 3-х уровневой. В случае, если получен некорректный номер блока, завершаем алгоритм (область индексных дескрипторов закончилась). Альтернативой проверки поля со ссылками может являться проверка содержимого поля «тип файла»: если оно является некорректным, то это так же означает, что область индексных дескрипторов закончилась.
2	Описать алгоритм определения размера файла в блоках по содержимому массива адресации блоков файла индексного дескриптора (модельной Unix системы). Считаем, что массив состоит из элементов беззнакового целого. Размер блока – 2048 байт. Считаем, что доступ к	Вначале рассчитываем, сколько четырехбайтовых чисел (unsigned int) поместится в одном блоке: tmp = 2048 / sizeof(unsigned int) = 2048 / 4 = 512.

№	Условие	Ответ
	блокам файловой системы осуществляется посредством использования внешней функции GetBlockFS, которая принимает в качестве параметра номер блока файловой системы, который нужно считать, а возвращает указатель на считанный блок.	Далее сначала рассматриваем первые 10 элементов массива адресации. Если встречаем 0, то останавливаемся. Рассматриваем 11-ый элемент. Если он равен нулю, то останавливаемся. Иначе при помощи функции GetBlockFS получаем указатель на следующий блок, содержащий 512 номеров блоков. Также их проверяем на ноль. Если не остановились, то переходим к 12-ому элементу, не забываем, что здесь уже косвенная адресация второго уровня (данный элемент ссылается на массив из 512 ссылок, каждая из которых ссылается на массив из 512 блоков файла). Далее, если не остановились, переходим к 13-ому элементу (где косвенная адресация уже третьего уровня).
3	Пусть дана файловая система Unix System V. Описать последовательность действий при запросе на получение свободного блока.	В первом блоке массива свободных блоков ищется ячейка со значением, не равным 0. Эта ячейка обнуляется, блок с соотв. номером выдается в ответ на запрос. Если же происходит обнуление последней ячейки блока (ссылка на следующий блок массива), то предварительно этот блок загружается в суперблок и становится первым блоком массива свободных блоков.

№	Условие	Ответ
4	Как работает системный вызов open(filename, openmode, flags)?	1. Открывает файл с именем filename, режимом доступа openmode, если openmode позволяет создание файла, то файл создается с правами flags. 2. устанавливается связь с индексным дескриптором, или создается новый ИД 3. добавляется новая запись в ТОФ ОС (указатель файла и ссылка на ИД) 4. добавляется запись в ТОФ процесса 5. индекс записи возвращается как дескриптор открытого файла

Задача 6

№	Условие	Ответ
1	Какова структура IP адреса класса C (описать все поля и их размеры)?	<код_класса><номер_сети><номер_компьютера_в_сети> <код_класса> – 110 (3 бита) <номер_компьютера_в_сети> – один байт <номер_сети> – оставшееся в IP адресе пространство (крайние левые три байта IP адреса без крайних левых трех битов)
2	Какова структура IP адреса класса B (описать все поля и их размеры)?	<код_класса><номер_сети><номер_компьютера_в_сети> <код_класса> – 10 (2 бита) <номер_компьютера_в_сети> – 2 байта

№	Условие	Ответ
		<номер_сети> – оставшееся в IP адресе пространство (крайние левые два байта IP адреса без крайних левых двух битов)
3	Какова структура IP адреса класса A (описать все поля и их размеры)?	<код_класса><номер_сети><номер_компьютера_в_сети> <код_класса> – 0 (1 бит) <номер_компьютера_в_сети> – 3 байта <номер_сети> – оставшееся в IP адресе пространство, 7 битов
4	Какова структура IP адреса класса D (описать все поля и их размеры)?	<код_класса><группа> <код_класса> – 1110 (4 бита) <группа> – оставшееся в IP адресе пространство (32 – 4 = 28 битов)
5	Сколько байтов в структуре IP-адреса класса C отводится под номер компьютера? Где они расположены?	<код_класса><номер_сети><номер_компьютера_в_сети> Один байт, крайний справа

Задача 7

№	Условие	Ответ
1	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все	**1** **01** **01** либо

№	Условие	Ответ

	системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	int fd[2];
	pipe(fd);
	char x[] = "01\n";
	if(fork()) {
	puts(x + 1);
	write(fd[1], x, 1);
	wait(NULL);
	}
	else {
	write(fd[1], &x[1], 1);
	read(fd[0], x, 1);
	read(fd[0], x+1, 1);
	}
	puts(x);
	return 0;
	}
		1
		10
		01

2	Что будет выведено на экран при выполнении фрагмента программы? Если допустимы несколько вариантов вывода, приведите все. Считаем, что все	**bf** либо

№	Условие	Ответ

	системные вызовы отрабатывают полностью и корректно – без отказов.
	char buf[5] = "abcf";
	int fd = creat("./prob.txt", 0777);
	write(fd, buf, 4);
	close(fd);
	fd = open("./prob.txt", O_RDONLY);
	fork();
	read(fd, buf, 2);
	printf("%c", buf[1]);
	exit(0);
		fb

3	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int main()
	{
	int fd[2];
	pipe(fd);
	char x[] = "ab\n";
	if(fork()) {
	puts(x + 1);
		b
		ab
		ab
		
		либо
		
		b
		ba
		ab

№	Условие	Ответ

	write(fd[1], x, 1);
	wait(0);
	} else {
	wait(0);
	write(fd[1], &x[1], 1);
	read(fd[0], x, 1);
	read(fd[0], x+1, 1);
	}
	puts(x);
	return 0;
	}	

4	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int main()
	{
	int fd[2];
	pipe(fd);
	char x[] = "01\n";
	if(fork()) {
	puts(x);
		01
		01
		
		либо
		
		01
		10

№	Условие	Ответ

	write(fd[1], x, 1);
	wait(NULL);
	} else {
	write(fd[1], &x[1], 1);
	read(fd[0], x, 1);
	read(fd[0], x+1, 1);
	puts(x);
	}
	return 0;
	}	

5	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int main()
	{
	int fd[2];
	pipe(fd);
	char x[] = "01\n";
	if(fork()) {
	write(fd[1], x, 1);
	wait(NULL);
		01
		01
		
		либо
		
		10
		01

№	Условие	Ответ

	} else {
	write(fd[1], &x[1], 1);
	read(fd[0], x, 1);
	read(fd[0], x+1, 1);
	}
	puts(x);
	return 0;
	}	

6	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int main()
	{
	int fd[2];
	pipe(fd);
	char x[] = "ab\n";
	if(fork()) {
	write(fd[1], x, 1);
	wait(NULL);
	}
	else {
		ab
		ab
		
		либо
		
		ba
		ab

№	Условие	Ответ

	write(fd[1], &x[1], 1);
	read(fd[0], x, 1);
	read(fd[0], x+1, 1);
	}
	puts(x);
	return 0;
	}	

7	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int main()
	{
	int fd[2];
	pipe(fd);
	char x[] = "20\n";
	if(fork()) {
	puts(x + 1);
	write(fd[1], x, 1);
	wait(NULL);
	}
	else {
		0
		20
		20
		
		либо
		
		0
		02
		20

№	Условие	Ответ

	write(fd[1], &x[1], 1);
	read(fd[0], x, 1);
	read(fd[0], x+1, 1);
	}
	puts(x);
	return 0;
	}	

8	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int main()
	{
	int fd[2];
	pipe(fd);
	char z = 3;
	if(fork())
	{
	printf("%i\n", z);
	z = z+1;
	write(fd[1],&z, 1);
	wait(NULL);
		3 \n 4 \n 4
		или
		6 \n 3 \n 4
		или
		3 \n 6 \n 4

№	Условие	Ответ

	}
	else
	{
	z = z+3;
	write(fd[1], &z, 1);
	read(fd[0], &z, 1);
	}
	printf("%i\n", z);
	return 0;
	}	

9	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int main()
	{
	int fd[2];
	pipe(fd);
	char x[] = "ab\n";
	if(fork()) {
	read(fd[0], x+1, 1);
	wait(NULL);
		ab
		ab
		a
		
		либо
		
		ab
		aa
		b

№	Условие	Ответ

	}
	else { puts(x);
	write(fd[1], x, 1);
	write(fd[1], &x[1], 1);
	read(fd[0], x+1, 1);
	puts(x);
	return 0;
	}
	puts(x+1);
	return 0;
	}	

10	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int main()
	{
	int fd[2];
	pipe(fd);
	char x[] = "123\n";
	if(fork()) {
	puts(x + 1);
		23
		313
		123
		
		либо
		
		23
		133
		123

№	Условие	Ответ

	write(fd[1], x+2, 1);
	wait(NULL);
	}
	else {
	write(fd[1], &x[0], 1);
	read(fd[0], x, 1);
	read(fd[0], x+1, 1);
	}
	puts(x);
	return 0;
	}	

11	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int main()
	{
	int fd[2];
	pipe(fd);
	char x[] = "qw\n";
	if(fork()) {// pid=A
	write(fd[1], x, 1);
		B qw
		A qw
		
		либо
		
		B wq
		A qw

№	Условие	Ответ

	wait(NULL);
	}
	else { //pid=B
	write(fd[1], &x[1], 1);
	read(fd[0], x, 1);
	read(fd[0], x+1, 1);
	}
	printf("%d " ,getpid());
	puts(x);
	return 0;
	}	

12	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int main() {
	int fd[2];
	pipe(fd);
	char x[] = "12\n";
	if(fork()) {
	puts(x + 1);
	write(fd[1], x, 1);
		2
		12
		12
		
		либо
		
		2
		21
		12

№	Условие	Ответ

	wait(NULL);
	}
	else {
	write(fd[1], &x[1], 1);
	read(fd[0], x, 1);
	read(fd[0], x+1, 1);
	}
	puts(x);
	return 0;
	}	

Задача 8

№	Условие	Ответ
1	Может ли одно и то же физическое устройство быть представлено в системе и как байт-ориентированное устройство и как блок-ориентированное? Обосновать.	Может. Регистрируются 2 файла устройств, связанных с данным устройством. Один файл – байт-ориентированное устройство (связано с соответствующим драйвером), другой – блок-ориентированное устройство
2	Привести 2 примера байт-ориентированных и блок-ориентированных устройств	Клавиатура, мышь, принтер
3	Привести 2 примера блок-ориентированных устройств	Флэш-накопитель, жесткий диск, накопитель на магнитной ленте

№	Условие	Ответ
4	Верно ли, что любое физическое устройство представлено в системе и как байт-ориентированное устройство и как блок-ориентированное? Обосновать.	Нет. Не для всех устройств оба варианта имеют смысл. Например, для датчика температуры блочное представление не нужно.

Задача 9

№	Условие	Ответ

1	Что будет выведено на экран, если PID изначально запущенного процесса равен 1277? Если возможны несколько вариантов – обосновать и привести все варианты. Предполагается, что все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int main()
	{
	int pid;
	if (fork() == 0) {
	printf ("PPID = %d \n", getppid());
	} else {
	exit(0);
	}
	}
		PPID=1277 или PPID=1
		в зависимости от того, как сработает планировщик

2	Что будет выведено на экран, если PID изначально запущенного процесса равен 1234? Если возможны	PPID=1234 или PPID=1 в зависимости от того, как сработает планировщик

№	Условие	Ответ

	несколько вариантов – обосновать и привести все варианты. Предполагается, что все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int
	main(void)
	{
	  if (fork() == 0) {
	  printf("PPID = %d\n", getppid());
	  } else {
	  exit(0);
	  }
	}	

3	Что будет выведено на экран, если PID изначально запущенного процесса равен 1242? Если возможны несколько вариантов – обосновать и привести все варианты. Предполагается, что все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int
	main(void)
	{
	  if (fork() == 0) {
	  printf("PPID = %d\n", getppid());
		PPID=1242 или PPID=1
		в зависимости от того, как сработает планировщик

№	Условие	Ответ

	wait(NULL);
	} else {
	exit(0);
	}
	}	

4	Что будет выведено на экран, если PID изначально запущенного процесса равен A, а PID запущенных процессов – B или C? Если возможны несколько вариантов – обосновать и привести все варианты. Предполагается, что все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int
	main()
	{
	if(fork()==0) // B
	if (fork()==0){ // C
	printf
	("PPID1=%d\n",getppid());
	}else{
	printf
	("PPID2=%d\n",getppid());
	exit(0);
	}
		PPID1=B
		PPID2=A
		или
		PPID2=A
		PPID1=B
		или
		PPID1=1
		PPID2=A
		или
		PPID2=A
		PPID1=1
		в зависимости от того, как сработает планировщик

№	Условие	Ответ

	else // A
	wait(NULL);
	}	

5	Что будет выведено на экран, если PID изначально запущенного процесса равен A, а PID запущенных процессов - B или C? Если возможны несколько вариантов – обосновать и привести все варианты. Предполагается, что все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int
	main()
	{
	if(fork()==0) // B
	if (fork()==0){ // C
	printf
	("PPID1=%d\n",getppid());
	}else{
	printf
	("PPID2=%d\n",getppid());
	wait(NULL); exit(0);
	}
	else // A
	exit(0);
		PPID1=B
		PPID2=A
		или
		PPID2=A
		PPID1=B
		или
		PPID1=B
		PPID2=1
		или
		PPID2=1
		PPID1=B
		в зависимости от того, как сработает планировщик

№	Условие	Ответ
	}	

6	Что будет выведено на экран, если PID изначально запущенного процесса равен 1148? Если возможны несколько вариантов – обосновать и привести все варианты. Предполагается, что все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int
	main(void)
	{
	  if (fork() == 0) {
	  printf("PPID = %d\n", getppid());
	  } else {
	  wait(NULL);
	  }
	}
		PPID=1148, вариант PPID=1 невозможен, так как отец дожидается завершения сына

7	Что будет выведено на экран, если PID изначально запущенного процесса равен 5431? Если возможны несколько вариантов – обосновать и привести все варианты. Предполагается, что все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. int main() {	PPID=5431 или PPID=1 в зависимости от того, как сработает планировщик

№	Условие	Ответ

	int pid;
	pid = getpid();
	if (fork() == 0){
	if(getppid()==pid){
	printf ("PPID = %d \n", pid);
	} else {
	printf ("PPID = %d \n", getppid());
	}
	} else {
	exit(0);
	}
	}	

Задача 10

№	Условие	Ответ
1	Может ли пользовательский процесс в Unix System V выполняться в режиме операционной системы? Обосновать ответ.	Да. При обращении к системным вызовам
2	Привести примеры библиотечных функций языка С, не содержащих в своей реализации системные вызовы	Большинство функций работы со строками (например, sscanf, strlen)
3	Чем отличается системный вызов от библиотечной функции? Вычеркнуть из списка все системные вызовы:	При обращении к системным вызовам процесс переходит в привилегированный режим, в котором выполняются нужные функции ядра ОС.

№	Условие	Ответ

	read(fd, buffer, N);
	scanf(“%d”, &i);
	sscanf(buffer, “%d”, &i);
	k = ftok(“/etc/passwd”, ‘A’);
	id = msgget(k, IPC_CREAT | 0666);
		Библиотечные функции работают в пользовательском режиме, обращаясь, если надо к системным вызовам.
		Библиотечные функции:
		scanf(“%d”, &i);
		sscanf(buffer, “%d”, &i);
		k = ftok(“/etc/passwd”, ‘A’);

4	Как работает системный вызов wait(int * status)?	• Процесс блокируется до завершения какого-либо потомка, если потомков нет (или всех уже дождались), то возвращается -1; • удаляет завершившийся процесс-потомок из таблицы процессов; • статус завершения потомка записывается в *status(если указатель ненулевой); • возвращает pid завершенного потомка.
5	Чем отличается системный вызов от библиотечной функции? Какие библиотечные функции ниже НЕ обращаются к системным вызовам? scanf(“%d”, &i); sscanf(buffer, “%d”, &i); k = ftok(“/etc/passwd”, ‘A’); d = sqrt(x);	При обращении к системным вызовам процесс переходит в привилегированный режим, в котором выполняются нужные функции ядра ОС. Библиотечные функции работают в пользовательском режиме, обращаясь, если надо к системным вызовам. Библиотечные функции, которые не обращаются к системным вызовам: sscanf(buffer, “%d”, &i);

№	Условие	Ответ
		d = sqrt(x);
6	В каких режимах будет работать процесс при выполнении функции printf()? Обосновать ответ.	Частично в пользовательском (подготовка данных для вывода), частично в привилегированном (собственно вывод через системный вызов write())
7	В каком режиме выполняется пользовательский процесс в Unix System V при обращении к системным вызовам?	В режиме операционной системы

Задача 11

№	Условие	Ответ
1	В системе клиент-сервер, реализованной с использованием сокетов, подключены и работают три клиентских процесса. Обосновать, какое минимальное количество сокетов может быть единовременно открыто у процесса-сервера в этом случае?	Один, т.к. сервер для каждого подключенного клиента может формировать отдельный процесс, после чего закрывать сокет, связанный с клиентом.
2	В системе клиент-сервер, реализованной с использованием сокетов, подключены и работают пять клиентских процессов. Обосновать, какое минимальное количество сокетов может быть единовременно открыто у процесса-сервера в этом случае?	Один, т.к. сервер для каждого подключенного клиента может формировать отдельный процесс, после чего закрывать сокет, связанный с клиентом.

№	Условие	Ответ
3	В системе клиент-сервер, реализованной с использованием сокетов, работает один серверный процесс, в котором открыто 3 сокета. Обосновать, какое минимальное количество клиентских процессов может единовременно работать в этом случае?	Ни одного клиентского процесса, т.к. сервер для каждого подключенного клиента может асинхронным образом обрабатывать соединения, не заводя для этого клиентский процесс.
4	В системе клиент-сервер, реализованной с использованием сокетов, работает один серверный процесс, в котором открыт 1 серверный и 3 клиентских сокета. Обосновать, какое минимальное количество клиентских процессов может единовременно работать в этом случае?	Ни одного клиентского процесса, т.к. сервер для каждого подключенного клиента может асинхронным образом обрабатывать соединения, не заводя для этого клиентский процесс.
5	В системе клиент-сервер, реализованной с использованием сокетов, подключены и работают три клиентских процесса. Обосновать, какое минимальное количество сокетов может быть единовременно открыто у процесса-сервера и его потомков в этом случае?	Один для сервера – для приема запросов на соединение от клиентов, один для каждого сыновнего процесса, занимающегося обслуживанием клиента. Итого 4.
6	Каким образом можно добиться того, чтобы в системе клиент-сервер (реализованной с использованием сокетов) при работе с пятью клиентскими процессами у процесса-сервера был бы открыт только один сокет?	Формировать отдельный процесс для каждого подключенного клиента, затем закрывать сокет, связанный с клиентом.

Задача 12

№	Условие	Ответ
1	Дана файловая система, имеющая организацию в виде связанного списка. Пусть ссылка в файловом блоке занимает 8 байтов, а размер файлового блока равен 1024 байтам. Пусть некоторый файл из данной файловой системы содержит текстовую информацию (последовательность байтов, содержащих коды символов). За какое минимальное количество обменов можно прочесть часть текста с 2047 символа до 3072 (считаем, что нумерация символов в тексте начинается с 1).	**Четыре обмена**
2	Дана файловая система, имеющая организацию в виде связанного списка. Пусть ссылка в файловом блоке занимает 4 байта, а размер файлового блока равен 1024 байтам. Пусть некоторый файл из данной файловой системы содержит текстовую информацию (последовательность байтов, содержащих коды символов). За какое минимальное количество обменов можно прочесть часть текста с 2047 символа до 5172 (считаем, что нумерация символов в тексте начинается с 1).	Содержательная информация в файловом блоке занимает: 1024 – 4 = 1020 байтов. Нам надо прочитать до 5172 байта: 5172 / 1020 = 5.071 блоков. Таким образом, нам надо пройти 6 блоков (то есть округляем 5.071 в большую сторону), то есть выполнить **6 обменов**.
3	Дана файловая система, имеющая организацию в виде связанного списка. Пусть ссылка в файловом блоке занимает 8 байтов, а размер файлового блока равен 2048 байтам. Пусть некоторый файл из данной	**3 обмена**

№	Условие	Ответ
	файловой системы содержит текстовую информацию (последовательность байтов, содержащих коды символов). За какое минимальное количество обменов можно прочесть часть текста с 2049 символа до 4090 включительно (считаем, что нумерация символов в тексте начинается с 0).	
4	Дана файловая система, имеющая организацию в виде связанного списка. Пусть ссылка в файловом блоке занимает 4 байта, а размер файлового блока равен 2048 байтам. Пусть некоторый файл из данной файловой системы содержит текстовую информацию (последовательность байтов, содержащих коды символов). За какое минимальное количество обменов можно прочесть часть текста с 100 символа до 2046 включительно (считаем, что нумерация символов в тексте начинается с 0).	**2 обмена**
5	Дана файловая система, имеющая организацию в виде связного списка. Пусть ссылка в файловом блоке занимает 16 байтов, а размер файлового блока равен 512 байтам. Пусть некоторый файл из данной файловой системы содержит текстовую информацию (последовательность байтов, содержащих коды символов). За какое минимальное количество обменов можно прочесть часть текста с 1135-го	**5 обменов**

№	Условие	Ответ
	символа до 2037-й (считаем, что нумерация символов в тексте начинается с 1).	
6	Дана файловая система, имеющая организацию в виде связного списка. Пусть ссылка в файловом блоке занимает 8 байтов, а размер файлового блока равен 512 байтам. Пусть некоторый файл из данной файловой системы содержит текстовую информацию (последовательность байтов, содержащих коды символов). За какое минимальное количество обменов можно прочесть часть текста с 1100-го символа до 1500-й (считаем, что нумерация символов в тексте начинается с 1).	**3 обмена**
7	Дана файловая система, имеющая организацию в виде связанного списка. Пусть ссылка в файловом блоке занимает 4 байта, а размер файлового блока равен 1024 байтам. Пусть некоторый файл из данной файловой системы содержит текстовую информацию (последовательность байтов, содержащих коды символов). За какое минимальное количество обменов можно прочесть часть текста с 3000 символа до 3050 (считаем, что нумерация символов в тексте начинается с 1).	**3 обмена** Первый считывает 1020 байтов, второй – еще 1020 (всего 2040), третий – еще 1020. Всего 3060 – это попадает в диапазон.
8	Дана файловая система, имеющая организацию в виде связанного списка. Пусть ссылка в файловом блоке	**5 обменов** Каждый обмен – по 1020 байтов.

№	Условие	Ответ
	занимает 4 байта, а размер файлового блока равен 1024 байтам. Пусть некоторый файл из данной файловой системы содержит текстовую информацию (последовательность байтов, содержащих коды символов). За какое минимальное количество обменов можно прочесть часть текста с 3000 символа до 5000 (считаем, что нумерация символов в тексте начинается с 1).	Четыре обмена – 4080 байтов (начало диапазона с 3000). Пятый – еще 1020 байтов. Всего 5100 – это попадает в диапазон
9	Дана файловая система, имеющая организацию в виде связанного списка. Пусть ссылка в файловом блоке занимает 8 байтов, а размер файлового блока равен 2048 байтов. Пусть некоторый файл из данной файловой системы содержит текстовую информацию (последовательность байтов, содержащих коды символов). За какое минимальное количество обменов можно прочесть часть текста с 2047 символа до 3072 (считаем, что нумерация символов в тексте начинается с 1).	**2 обмена**
10	Дана файловая система, имеющая организацию в виде связанного списка. Пусть ссылка в файловом блоке занимает 4 байта, а размер файлового блока равен 512 байтам. Пусть некоторый файл из данной файловой системы содержит текстовую информацию (последовательность байтов, содержащих коды	**9 обменов**

№	Условие	Ответ
	символов). За какое минимальное количество обменов можно прочесть часть текста с 4092 символа до 4108 (считаем, что нумерация символов в тексте начинается с 1).	
11	Дана файловая система, имеющая организацию в виде связанного списка. Пусть ссылка в файловом блоке занимает 8 байтов, а размер файлового блока равен 1024 байтам. Пусть некоторый файл из данной файловой системы содержит текстовую информацию (последовательность байтов, содержащих коды символов). За какое минимальное количество обменов можно прочесть часть текста с 2000 символа до 3000 включительно (считаем, что нумерация символов в тексте начинается с 0).	**3 обмена**
12	Дана файловая система, имеющая организацию в виде связанного списка. Пусть ссылка в файловом блоке занимает 4 байта, а размер файлового блока равен 2048 байтам. Пусть некоторый файл из данной файловой системы содержит текстовую информацию (последовательность байтов, содержащих коды символов). За какое минимальное количество обменов можно прочесть часть текста с 3000 символа до 7 000 (считаем, что нумерация символов в тексте начинается с 0).	**4 обмена**

Задача 13

№	Условие	Ответ
1	При сверке целостности файловой системы i-й элемент таблицы занятых блоков равен 3. А i-й элемент таблицы свободных блоков равен 5. Описать последовательность действий, восстанавливающих системную информацию файловой системы.	Находим 3 индексных дескриптора, содержащих блок с номером i, делаем копию соответствующих файлов, удаляем 3 оригинальных файла, переименовываем копии файлов в имена исходных файлов, перевосстанавливаем таблицу занятых блоков, обнуляем i-ю запись в таблице свободных.
2	Дать краткое описание основных шагов алгоритма восстановления списка свободных индексных дескрипторов файловой системы System V.	Цикл по области индексных дескрипторов. Индексный дескриптор считается свободным, если его поле «количество ссылок из каталогов файловой системы» равно 0.
3	Пусть в файловой системе используется модель учёта свободных блоков на основе битовых массивов. Сколько блоков ФС займёт этот битовый массив для жёсткого диска объёмом в 8 Гбайт, если размер блока равен 2 Кбайт?	Диск = (8*1024*1024) / 2 = 2^22 блоков; 2^22 бит = 2^8 * 2 Кбайт = 2^8 блоков

Задача 14

№	Условие	Ответ
1	Перечислить основные шаги инкрементального архивирования файлов.	Создание цепочки архивов: 1. Создаем мастер копию архива – копия всех архивируемых файлов.

№	Условие	Ответ
		2. По расписанию создаем копии «изменений» – копия, в которой сохранены файлы созданные или измененные с момента предыдущего архивирования.
2	В системе, в различных процессах единовременно N кратно открыт файл с именем Name – в существующих в системе процессах имеется N файловых дескрипторов, связанных с файлом Name. Из которых K файловых дескрипторов являются унаследованными. Какое количество записей, связанных с данным файлом, имеется в Таблице файлов операционной системы?	Получается, унаследованы k файловых дескрипторов. Значит, в отце открыли файл k раз, получилось k дескрипторов – k записей в ТФ. Далее в сыновьях унаследовались k открытых файлов, при этом сами по себе новые записи в ТФ не появлялись. Файл уже открыт 2*k раз. Осталось открыть N-2*k раз, уже после fork. Открываем, получаем еще N-2*k новых записей в ТФ. И в сумме: k+N-2k = **N-k**.
3	Если файл не является символической ссылкой, то где именно хранится ссылка на индексный дескриптор этого файла?	В файле каталога.
4	Перечислите все ситуации, в которых системный вызов fork() может вернуть -1.	• Системе не хватает ресурсов для размещения нового процесса; • Превышен лимит текущего процесса на создание потомков.
5	Где именно хранится имя файла в файловой системе?	В файле каталога.

№	Условие	Ответ
6	Перечислите все ситуации, в которых системный вызов waitpid(pid, NULL, flags) может вернуть -1.	• У процесса нет потомков, которых надо дожидаться (либо вообще не было, либо всех дождались); • Неверный pid; • Неверные флаги.
7	Каково основное преимущество инкрементального архивирования файлов?	При изменении файлов добавляют только различия, не надо все заново архивировать.
8	Перечислить достоинства и недостатки компрессии при архивировании.	Достоинства: выигрыш в объеме резервной копии Недостатки: компрессия чувствительна к потере информации. Потеря/добавление одного бита может повлечь за собой порчу всего архива.

Задача 15

№	Условие	Ответ
1	Пусть дан 32-разрядный компьютер, в котором реализована двухуровневая таблица страниц. Размер страницы 2048 байтов. Размер каждой таблицы второго уровня равен 1024 записи. Сколько записей содержит «внешняя таблица страниц»?	2048
2	Пусть в 32-х разрядном компьютере используется страничная память с двухуровневой таблицей страниц. Размер страницы 4096 байт. Таблица страниц первого уровня (внешняя) содержит 8192	Виртуальный адрес – это комбинация номера страницы первого уровня, номера страницы второго уровня и смещения в странице, всего 32 бита.

№	Условие	Ответ
	записи. Определить размер каждой таблицы второго уровня.	$4096 = 2^{12}$, следовательно под смещение в странице отводится 12 бит. $8192 = 2^{13}$, следовательно под номер страницы первого уровня отводится 13 бит. Под номер страницы второго уровня остается 32-12-13=7 бит. Следовательно, размер каждой таблицы второго уровня равен 2^7=128.
3	Пусть дан 32 разрядный компьютер, в котором реализована двухуровневая таблица страниц. Размер страницы 4096 байтов. Размер каждой таблицы второго уровня равен 2048 записи. Сколько записей содержит «внешняя таблица страниц»?	512
4	Пусть дан 16 разрядный компьютер, в котором реализована двухуровневая таблица страниц. Размер страницы 64 байта. Размер каждой таблицы второго уровня равен 64 записи. Сколько записей содержит «внешняя таблица страниц»?	16
5	Пусть дан 32 разрядный компьютер, в котором реализована двухуровневая таблица страниц. Размер страницы 2048 байтов. Размер каждой таблицы второго уровня равен 1024 записи. Сколько записей содержит «внешняя таблица страниц»?	2048

№	Условие	Ответ
6	Пусть дан 32-разрядный компьютер, в котором реализована двухуровневая таблица страниц. Размер страницы 4096 байтов. Каждая таблица второго уровня содержит 256 записей. Сколько записей содержит «внешняя таблица страниц»?	2 ^ (32 – log 4096 – log 256) = 2 ^ (32 – 12 – 8) = 2 ^ 12 = 4096
7	Пусть дан 32-разрядный компьютер, в котором реализована двухуровневая таблица страниц. Размер страницы 1024 байтов. Каждая таблица второго уровня состоит из 512 записей. Сколько записей содержит «внешняя таблица страниц»?	2 ^ (32 – log 1024 – log 512) = 2 ^ (32 – 10 – 9) = 2 ^ 13 = 8 * 1024 = 8192
8	Пусть дан 32 разрядный компьютер, в котором реализована двухуровневая таблица страниц. Размер страницы 4096 байтов. Размер каждой таблицы второго уровня равен 512 записей. Сколько записей содержит «внешняя таблица страниц»?	2048
9	Пусть дан 64 разрядный компьютер, в котором реализована трехуровневая таблица страниц. Размер страницы 4096 байтов. Размер каждой таблицы второго уровня и третьего уровней равен 65536 записей. Сколько записей содержит «внешняя таблица страниц»?	1024*1024 записей
10	Пусть дан 32 разрядный компьютер, в котором реализована двухуровневая таблица страниц. Размер страницы 1024 байтов. Размер каждой таблицы	2048

№	Условие	Ответ
	второго уровня равен 2048 записи. Сколько записей содержит «внешняя таблица страниц»?	
11	Пусть дан 32 разрядный компьютер, в котором реализована трёхуровневая таблица страниц. Размер страницы 2048 байтов. Размер каждой таблицы третьего уровня равен 1024 записи. Размер каждой таблицы второго уровня равен 32 записи. Сколько записей содержит таблица первого уровня?	64 записи
12	Пусть дан 32 разрядный компьютер, в котором реализована трёхуровневая таблица страниц. Размер страницы 1024 байтов. Размер каждой таблицы третьего уровня равен 2048 записей. Размер таблицы первого уровня равен 64 записи. Сколько записей содержит каждая таблица второго уровня?	32 записи
13	Пусть дан 32 разрядный компьютер, в котором реализована двухуровневая таблица страниц. Размер страницы 1024 байта. Размер каждой таблицы второго уровня равен 1024 записи. Сколько записей содержит «внешняя таблица страниц»?	(4096 = 2^12) 32 = 10 (2^10 =1024 – смещение в странице) + 10 (2^10=1024 – смещение в табл 2 уровня) + 12 (осталось для внешней таблицы)
14	Пусть дан 32 разрядный компьютер, в котором реализована двухуровневая таблица страниц. Размер страницы 2048 байтов. Размер каждой таблицы	4096

№	Условие	Ответ
	второго уровня равен 512 записи. Сколько записей содержит «внешняя таблица страниц»?	

Задача 16

№	Условие	Ответ
1	Дать формальное описание семафора Дейкстры, который может использоваться для реализации взаимного исключения.	Дать определение семафора, у которого максимальное значение равно 1
2	Дать формальное описание семафора Дейкстры, который может использоваться для реализации одновременного доступа к ресурсу не более 4-х процессов.	Дать определение семафора, у которого максимальное значение равно 4
3	Привести схему взаимного исключения процессов с помощью двоичного семафора Дейкстры.	Семафор в начальном состоянии должен быть открыт (=1). Схема: P (закрыть семафор) – критическая секция – V (открыть семафор)

4	Привести схему взаимного исключения процессов с помощью семафоров IPC.	Семафор (semid) в начальном состоянии должен быть открыт (=1). Схема:
		struct sembuf P = {0, -1, 0};
		struct sembuf V = {0, 1, 0};
		semop(semid, &P, 1); //(закрыть семафор)
		– критическая секция –
		semop(semid, &V,1); // (открыть семафор)

№	Условие	Ответ

5	Можно ли реализовать семафор Дейкстры через обычную целую переменную и активное ожидание, например:
	int sem =1;
	…
	while (sem == 0); // ждем пока семафор поднимут
	sem=0; // входим в критическую секцию
	//работаем с разделяемым ресурсом
	sem=1; // выходим из критической секции
		Нет, отсутствует атомарность: между проверкой и установкой семафора может произойти переключение на другой процесс

6	Дать формальное описание семафора Дейкстры, который может использоваться для реализации единовременного доступа к ресурсу не более чем 2-х процессов.	Дать определение семафора, у которого максимальное значение равно 2
7	Дать описание операций, которые могут выполняться над семафором Дейкстры.	Операция down(S): проверяет значение семафора S, если оно больше нуля, то уменьшает его на 1, иначе процесс блокируется (связанная с процессом операция down считается незавершенной). Операция up(S): увеличивает значение семафора на 1. Если в системе присутствуют процессы, блокированные ранее операцией down на этом семафоре, один из них разблокируется и завершает выполнение операции down.
8	Дать формальное описание семафора Дейкстры, который может использоваться для реализации	Дать определение семафора, у которого максимальное значение равно 1.

№	Условие	Ответ
	взаимного исключения. Какие средства межпроцессного взаимодействия (кроме собственно семафоров) могут быть использованы для реализации взаимного исключения и как?	Каналы через блокировку на чтение из пустого канала. Очереди сообщений через блокировку на чтение сообщений нужного типа. Сигналы через блокировку на ожидании сигнала.

Задача 17

№	Условие	Ответ
1	Дана файловая система, имеющая архитектуру аналогичную fs5. Пусть размер ссылки на блок файловой системы – 4 байта; размер блока 32 байта. Какой предельный размер файла в блоках могут иметь файлы в такой файловой системе (указать число)?	$10+8+8^2+8^3 = 594$
2	Дана файловая система, имеющая архитектуру аналогичную fs5. Пусть размер ссылки на блок файловой системы – 8 байт; размер блока 32 байта. Какой предельный размер файла (в байтах) могут иметь файлы в такой файловой системе (указать число)?	$(10+4+4^2+4^3) * 32 = 3008$
3	Дана файловая система, имеющая архитектуру аналогичную fs5. Пусть размер ссылки на блок файловой системы – 8 байтов; размер блока 32 байта.	$10+4+4^2+4^3 = 94$

№	Условие	Ответ
	Какой предельный размер файла в блоках могут иметь файлы в такой файловой системе (указать число)?	
4	Дана файловая система, имеющая архитектуру аналогичную fs5. Пусть размер ссылки на блок файловой системы – 8 байтов; размер блока 64 байта. Какой предельный размер файла в блоках могут иметь файлы в такой файловой системе (указать число)?	$10+8+8^2+8^3 = 594$
5	Дана файловая система, архитектура которой аналогична fs5. Пусть размер ссылки на блок файловой системы – 4 байта; размер блока 64 байтов. Найдите предельный размер файла в такой файловой системе (указать константное выражение) в байтах.	64/4 = 16, $64 * (10+16+16^2+16^3)$
6	Дана файловая система, имеющая архитектуру аналогичную fs5. Пусть размер ссылки на блок файловой системы – 8 байтов; размер блока 2048 байтов. Какой предельный размер файла в блоках могут иметь файлы в такой файловой системе (указать константное выражение)?	$10+256+256^2+256^3$
7	Дана файловая система, имеющая архитектуру аналогичную fs5. Пусть размер ссылки на блок файловой системы – 4 байта; размер блока 2048 байтов. Какой предельный размер файла в блоках	$10+512+512^2+512^3$

№	Условие	Ответ
	могут иметь файлы в такой файловой системе (указать константное выражение)?	
8	Дана файловая система, имеющая архитектуру аналогичную fs5. Пусть размер ссылки на блок файловой системы – 2 байта; размер блока 64 байта. Какой предельный размер файла в блоках могут иметь файлы в такой файловой системе (указать константное выражение)?	$10+32+32^2+32^3$
9	Дана файловая система, имеющая архитектуру аналогичную fs5. Пусть размер ссылки на блок файловой системы – 4 байта; размер блока 64 байта. Какой предельный размер файла в блоках могут иметь файлы в такой файловой системе (указать число)?	$10+16+16^2+16^3 = 4378$
10	Дана файловая система, имеющая архитектуру аналогичную fs5. Пусть размер ссылки на блок файловой системы – 8 байтов; размер блока 128 байт. Какой предельный размер файла в блоках могут иметь файлы в такой файловой системе (указать число)?	$10+16+16^2+16^3 = 4378$

Задача 18

№	Условие	Ответ
1	Пусть в некотором компьютере реализована страничная организация памяти, таблица страниц операционной системы имеет размер 4096 записей. Размер страницы 512 байтов. Определите разрядность виртуального адреса для данного случая.	21 разряд
2	Пусть в некотором компьютере реализована страничная организация памяти с использование инвертированной таблицы страниц, состоящей из 300 записей. Размер виртуальной страницы ОЗУ 2048 байтов. Каков объем физической памяти этого компьютера?	300 х 2048 байтов
3	Пусть в некотором компьютере реализована страничная организация памяти, таблица страниц операционной системы имеет размер 8192 записи. Размер страницы 1024 байта. Определите разрядность виртуального адреса для данного случая.	23 разряда
4	Пусть в некотором компьютере реализована страничная организация памяти, таблица страниц операционной системы имеет размер 2048 записей. Размер страницы 512 байтов. Определите разрядность виртуального адреса для данного случая.	20 разрядов
5	Пусть в некотором компьютере реализована страничная организация памяти, таблица страниц операционной системы имеет размер 2048 записей.	log 2048 + log 4096 = 23 разряда

№	Условие	Ответ
	Размер страницы 4096 байтов. Определите разрядность виртуального адреса для данного случая.	
6	Пусть в некотором компьютере реализована страничная организация памяти, таблица страниц операционной системы имеет размер 512 записей. Размер страницы 1024 байта. Определите разрядность виртуального адреса для данного случая.	log 512 + log 1024 = 19 разрядов
7	Пусть в некотором компьютере реализована страничная организация памяти, таблица страниц операционной системы имеет размер 16384 записи. Размер страницы 4096 байтов. Определите разрядность виртуального адреса для данного случая.	14 битов под номер страницы, 12 битов – под смещение – итого 26 битов.
8	Пусть в некотором компьютере реализована страничная организация памяти, таблица страниц операционной системы имеет размер 2048 записей. Размер страницы 256 байтов. Определите разрядность виртуального адреса для данного случая.	19 разрядов
9	Пусть в некотором компьютере реализована страничная организация памяти, таблица страниц операционной системы имеет размер 4096 записей. Разрядность виртуального адреса равна 22. Определите размер страницы для данного случая.	**10 разрядов** 2^12 =4096 22-12=10
10	Пусть в некотором компьютере реализована страничная организация памяти, таблица страниц	Log2(8192 * 4096) = 25 разрядов

№	Условие	Ответ
	операционной системы имеет размер 8192 записи. Размер страницы 4096 байт. Определите разрядность виртуального адреса для данного случая.	
11	Пусть в некотором компьютере реализована страничная организация памяти, таблица страниц операционной системы имеет размер 4096 записей. Размер страницы 1024 байтов. Определите разрядность виртуального адреса для данного случая.	22 разряда

Задача 19

№	Условие	Ответ
1	Для каких целей в рассмотренной в курсе модели обработки прерываний введена блокировка прерываний?	Для того, чтобы в момент сохранения точки и контекста прерывания не пришло другое прерывание, и не произошла потеря информации о точке первого прерывания.
2	Всегда ли в устройствах, работающих по протоколу TCP/IP, поддерживаются все 4 уровня взаимодействия (протокола)? Ответ обосновать.	Нет, не всегда (например, в шлюзах реализованы только первые два уровня, поскольку они просто передают информацию, и уровни более высокого уровня представления в них не нужны).
3	Чем отличаются длинные и короткие прерывания? Дать пример длинного прерывания.	При коротком прерывании не происходит смена контекста выполняемого процесса, поэтому достаточно только малого упрятывания информации о выполняемой программе (флаги,

№	Условие	Ответ
		регистры процессора). Длинное прерывание – от контроллера прямого доступа в память
4	Чем отличаются длинные и короткие прерывания? Дать пример короткого прерывания.	При коротком прерывании не происходит смена контекста выполняемого процесса, поэтому достаточно только малого упрятывания информации о выполняемой программе (флаги, регистры процессора). Короткое прерывание – от клавиатуры.
5	Для каких целей в рассмотренной в курсе модели обработки прерываний введена блокировка прерываний? В каком из средств межпроцессного взаимодействия используется (может быть использован) похожий механизм?	Для того, чтобы в момент сохранения точки и контекста прерывания не пришло другое прерывание, и не произошла потеря информации о точке первого прерывания. Такой же механизм может использоваться при обработке сигналов – пока обрабатывается один сигнал, доставка других сигналов может быть заблокирована.

Задача 20

№	Условие	Ответ
1	Что будет выведено на экран? Если возможны несколько вариантов – привести все.	32 либо 312

№	Условие	Ответ

	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	pid_t pid;
	int fd[2];
	int x = 3;
	pipe(fd);
	if((pid = fork()) > 0) {
	read(fd[0], &x, sizeof(int));
	kill(pid, SIGKILL);
	wait(NULL);
	} else {
	printf("%d", x);
	x = 2;
	write(fd[1], &x, sizeof(int));
	x = 1;
	}
	printf("%d", x);
	return 0;
	}	

№	Условие	Ответ

2	Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	int fd[2];
	pipe(fd);
	char x = 'a';
	if(fork()) {
	puts("w");
	write(fd[1], &x, 1);
	wait(NULL);
	puts("c");
	} else {
	close(fd[1]);
	kill(getppid(), SIGKILL);
	read(fd[0], &x, 1);
	puts("b");
	}
	return 0;
	}
		b
		либо
		w
		b
		либо
		w
		b
		c
		Третий случай появляется, поскольку сигнал может идти долго.

№	Условие	Ответ

3	Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	pid_t pid;
	int fd[2];
	int x = 3;
	pipe(fd);
	if((pid = fork()) > 0) {
	read(fd[0], &x, sizeof(int));
	kill(pid, SIGKILL);
	wait(NULL);
	} else {
	printf("%d", x);
	x = 1;
	write(fd[1], &x, sizeof(int));
	x = 2;
	}
	printf("%d", x);
	return 0;
		31 либо 321

№	Условие	Ответ
	}	

4	Что будет выведено на экран? Если возможны несколько вариантов, привести все. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	int main(void)
	{
	pid_t pid;
	int fd[2];
	char x[] = “abc”;
	pipe(fd);
	if ((pid = fork()) == 0) {
	write(1, x, 1); x[0] = ‘f’;
	write(fd[1], &x, 2); x[0] = ‘s’;
	} else {
	read(fd[0], &x, 2);
	kill(pid, SIGKILL);
	wait(NULL);
	}
	write(1, x, 1);
	return 0;
	}
		asf
		либо
		af

5	Что будет выведено на экран? Если возможны несколько вариантов – привести все.	2 либо

№	Условие	Ответ

	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	pid_t pid;
	int fd[2];
	int x = 3;
	pipe(fd);
	if((pid = fork()) > 0) {
	read(fd[0], &x, sizeof(int));
	kill(pid, SIGKILL);
	wait(NULL);
	} else {
	x = 2;
	write (fd[1], &x, sizeof(int));
	printf("%d", x);
	x = 1;
	}
	printf("%d", x);
	return 0;
	}
		22
		либо
		212

№	Условие	Ответ

6	Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	pid_t pid;
	int fd[2];
	int x = 3;
	pipe(fd);
	if((pid = fork()) > 0) {
	write(fd[0], &x, sizeof(int));
	kill(pid, SIGKILL);
	wait(NULL);
	} else {
	read(fd[1], &x, sizeof(int));
	printf("%d", x);
	x = 1;
	}
	printf("%d", x);
	return 0;
	}
		3
		либо
		33
		либо
		313

№	Условие	Ответ

7	Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	pid_t pid;
	int fd[2];
	int x = 3;
	pipe(fd);
	if((pid = fork()) > 0) {
	read(fd[0], &x, sizeof(int));
	kill(pid, SIGKILL);
	} else {
	printf("%d", x); x = 2;
	write(fd[1], &x, sizeof(int)); x = 1;
	}
	printf("%d", x);
	return 0;
	}
		32 либо 312 либо 321

8	Что будет выведено на экран? Если возможны несколько вариантов – привести все.	13 либо 123

№	Условие	Ответ

	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	pid_t pid;
	int fd[2];
	int x =1;
	pipe(fd);
	if((pid = fork()) > 0) {
	read(fd[0], &x, sizeof(int));
	kill(pid, SIGKILL);
	wait(NULL);
	} else {
	printf("%d", x); x = 3;
	write(fd[1], &x, sizeof(int)); x = 2;
	}
	printf("%d", x);
	return 0;
	}	

9	Что будет выведено на экран? Если возможны несколько вариантов – привести все.	ab либо acb

№	Условие	Ответ

	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	pid_t pid;
	int fd[2];
	char c = 'a';
	pipe(fd);
	if((pid = fork()) > 0) {
	read(fd[0], &c, 1);
	kill(pid, SIGKILL);
	wait(NULL);
	} else {
	putchar(c);
	c = 'b';
	write(fd[1], &c, 1);
	c = 'c';
	}
	putchar(c);
	return 0;
	}	

№	Условие	Ответ

10	Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	pid_t pid;
	int fd[2];
	int x = 9;
	pipe(fd);
	if((pid = fork()) > 0) {
	printf("%d", x - 1);
	read(fd[0], &x, sizeof(int));
	kill(pid, SIGKILL);
	wait(NULL);
	} else {
	x = 6;
	printf("%d", x);
	x = 3;
	write(fd[1], &x, sizeof(int));
	x = 7;
	}
		863
		683
		
		8673
		6873
		
		6783

№	Условие	Ответ

	printf("%d", x);
	return 0;
	}
	Поясните свой ответ.	

11	Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	pid_t pid;
	int fd[2];
	int x = 7;
	pipe(fd);
	if((pid = fork()) > 0) {
	read(fd[0], &x, sizeof(int));
	kill(pid, SIGKILL);
	wait(NULL);
	} else {
	printf("%d", x);
	x = 5;
	write(fd[1], &x, sizeof(int));
		75 либо 735

№	Условие	Ответ

	x = 3;
	}
	printf("%d", x);
	return 0;
	}	

Задача 21

№	Условие	Ответ
1	Что будет выведено на экран в результате работы фрагмента программы? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. msgId – идентификатор существующей пустой очереди сообщений. struct { long type; char data[1]; } msg;	bad

№	Условие	Ответ

	……..
	msg.type = 1; msg.data[0] = 'a'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 'b'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 'c'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 1; msg.data[0] = 'd'; msgsnd(msgId, &msg, 1, 0);
	msgrcv(msgId, &msg, 1, 2, 0); putchar(msg.data[0]);
	
	msgrcv(msgId, &msg, 1, 0, 0); putchar(msg.data[0]);
	msgrcv(msgId, &msg, 1, 1, 0); putchar(msg.data[0]);
	……	

2	Что будет выведено на экран в результате работы фрагмента программы? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. msgId – идентификатор существующей пустой очереди сообщений.	21b

№	Условие	Ответ

	struct
	{
	  long type;
	  char data[1];
	} msg;
	……..
	msg.type = 1; msg.data[0] = '1'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = '2'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 'a'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 1; msg.data[0] = 'b'; msgsnd(msgId, &msg, 1, 0);
	msgrcv(msgId, &msg, 1, 2, 0); putchar(msg.data[0]);
	msgrcv(msgId, &msg, 1, 0, 0); putchar(msg.data[0]);
	msgrcv(msgId, &msg, 1, 1, 0); putchar(msg.data[0]);
	……	

3	Что будет выведено на экран в результате работы фрагмента программы? Если возможны несколько вариантов – привести все. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.	yxz

№	Условие	Ответ

	msgId – идентификатор существующей пустой очереди сообщений.
	struct
	{
	  long type;
	  char data[1];
	} msg;
	……..
	msg.type = 1; msg.data[0] = 'x'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 'y'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 'z'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 1; msg.data[0] = 't'; msgsnd(msgId, &msg, 1, 0);
	msgrcv(msgId, &msg, 1, 2, 0); write(1, msg.data[0], 1);
	msgrcv(msgId, &msg, 1, 1, 0); write(1, msg.data[0], 1);
	msgrcv(msgId, &msg, 1, 0, 0); write(1, msg.data[0], 1);
	……	

4	Что будет выведено на экран в результате работы фрагмента программы? Если возможны несколько вариантов – привести все.	qpr

№	Условие	Ответ

	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	
	msgId – идентификатор существующей пустой очереди сообщений.
	struct
	{
	  long type;
	  char data[1];
	} msg;
	……..
	msg.type = 1; msg.data[0] = 'p'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 'q'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 1; msg.data[0] = 'r'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 's'; msgsnd(msgId, &msg, 1, 0);
	msgrcv(msgId, &msg, 1, 2, 0); write(1, msg.data[0], 1);
	msgrcv(msgId, &msg, 1, 0, 0); write(1, msg.data[0], 1);
	msgrcv(msgId, &msg, 1, 1, 0); write(1, msg.data[0], 1);
	……	

№	Условие	Ответ

5	Что будет выведено на экран в результате работы фрагмента программы? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	
	msgId – идентификатор существующей пустой очереди сообщений.
	struct {
	  long type;
	  char data[1];
	} msg;
	……..
	msg.type = 1; msg.data[0] = 'a'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 'b'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 'c'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 1; msg.data[0] = 'd'; msgsnd(msgId, &msg, 1, 0);
	msgrcv(msgId, &msg, 1, 0, 0); putchar(msg.data[0]);
		adb

№	Условие	Ответ

	msgrcv(msgId, &msg, 1, 1, 0); putchar(msg.data[0]);
	msgrcv(msgId, &msg, 1, 2, 0); putchar(msg.data[0]);
	……	

6	Что будет выведено на экран в результате работы фрагмента программы? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	
	msgId – идентификатор существующей пустой очереди сообщений.
	struct {
	long type;
	char data[1];
	} msg;
	……..
	msg.type = 1; msg.data[0] = 'a'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 'b'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 'c'; msgsnd(msgId, &msg, 1, 0);
		bca

№	Условие	Ответ

	msg.type = 1; msg.data[0] = 'd'; msgsnd(msgId, &msg, 1, 0);
	msgrcv(msgId, &msg, 1, 2, 0); putchar(msg.data[0]);
	msgrcv(msgId, &msg, 1, 2, 0); putchar(msg.data[0]);
	msgrcv(msgId, &msg, 1, 0, 0); putchar(msg.data[0]);
	……	

7	Что будет выведено на экран в результате работы фрагмента программы? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	
	msgId – идентификатор существующей пустой очереди сообщений.
	struct
	{
	long type;
	char data[1];
	} msg;
	……..
	msg.type = 1; msg.data[0] = '1'; msgsnd(msgId, &msg, 1, 0);
		214

№	Условие	Ответ

	msg.type = 2; msg.data[0] = '2'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = '3'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 1; msg.data[0] = '4'; msgsnd(msgId, &msg, 1, 0);
	msgrcv(msgId, &msg, 1, 2, 0); putchar(msg.data[0]);
	msgrcv(msgId, &msg, 1, 0, 0); putchar(msg.data[0]);
	msgrcv(msgId, &msg, 1, 1, 0); putchar(msg.data[0]);
	……	

8	Что будет выведено на экран в результате работы фрагмента программы? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. msgId – идентификатор существующей пустой очереди сообщений. struct { long type; char data[1];	abc

№	Условие	Ответ

	} msg;
	……..
	msg.type = 1; msg.data[0] = 'b'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 'a'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = ‘d'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 1; msg.data[0] = ‘c'; msgsnd(msgId, &msg, 1, 0);
	msgrcv(msgId, &msg, 1, 2, 0); putchar(msg.data[0]);
	msgrcv(msgId, &msg, 1, 0, 0); putchar(msg.data[0]);
	msgrcv(msgId, &msg, 1, 1, 0); putchar(msg.data[0]);
	……	

9	Что будет выведено на экран в результате работы фрагмента программы? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. msgId – идентификатор существующей пустой очереди сообщений.	cae

№	Условие	Ответ

	struct
	{
	  long type;
	  char data[1];
	} msg;
	……..
	msg.type = 1; msg.data[0] = 'a'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 'b'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 3; msg.data[0] = 'c'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 'd'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 1; msg.data[0] = 'e'; msgsnd(msgId, &msg, 1, 0);
	msgrcv(msgId, &msg, 1, 3, 0); putchar(msg.data[0]);
	msgrcv(msgId, &msg, 1, 0, 0); putchar(msg.data[0]);
	msgrcv(msgId, &msg, 1, 1, 0); putchar(msg.data[0]);
	……	

10	Что будет выведено на экран в результате работы фрагмента программы? Если возможны несколько вариантов – привести все.	ehll

№	Условие	Ответ

	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	
	msgId – идентификатор существующей пустой очереди сообщений.
	struct
	{
	long type;
	char data[1];
	} msg;
	……..
	msg.type = 1; msg.data[0] = 'h'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 'e'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 1; msg.data[0] = 'l'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 'l'; msgsnd(msgId, &msg, 1, 0);
	msg.type = 2; msg.data[0] = 'o'; msgsnd(msgId, &msg, 1, 0);
	msgrcv(msgId, &msg, 1, 2, 0); putchar(msg.data[0]);	

№	Условие	Ответ

	msgrcv(msgId, &msg, 1, 0, 0); putchar(msg.data[0]);
	msgrcv(msgId, &msg, 1, 1, 0); putchar(msg.data[0]);
	msgrcv(msgId, &msg, 1, 0, 0); putchar(msg.data[0]);
	……	

11	Что будет выведено на экран в результате работы фрагмента программы? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	
	msgId – идентификатор существующей пустой очереди сообщений.
	struct
	{
	long type;
	char data[1];
	} msg;
	……..
	msg.type =10; msg.data[0] = 'a'; msgsnd(msgId, &msg, 1, 0);
	msg.type =15; msg.data[0] = 'b'; msgsnd(msgId, &msg, 1, 0);
		cab

№	Условие	Ответ

	msg.type = 2; msg.data[0] = 'c'; msgsnd(msgId, &msg, 1, 0);
	msg.type =15; msg.data[0] = 'd'; msgsnd(msgId, &msg, 1, 0);
	msgrcv(msgId, &msg, 1, -10, 0); putchar(msg.data[0]);
	msgrcv(msgId, &msg, 1, 0, 0); putchar(msg.data[0]);
	msgrcv(msgId, &msg, 1, 15, 0); putchar(msg.data[0]);
	……	

Задача 22

№	Условие	Ответ

1	Что будет выведено на экран? Если возможны несколько вариантов– привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	char c;
	int fd[2], fd2[2];
	pipe(fd); pipe(fd2);
	if(fork() == 0) {
		acdbf либо adcbf либо dacbf

№	Условие	Ответ

	write(fd[1], &c, 1);
	putchar('d');
	read(fd2[0], &c, 1);
	putchar('b');
	exit(0);
	}
	putchar('a'); read(fd[0], &c, 1);
	putchar('c'); write(fd2[1], &c, 1);
	wait(NULL);
	putchar('f'); return 0;
	}	

2	Что будет выведено на экран? Если возможны несколько вариантов– привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	char c = 'e';
	int fd[2], fd2[2];
	pipe(fd); pipe(fd2);
	if(fork() == 0) {
	putchar('d');
		adcbf либо dacbf

№	Условие	Ответ

	write(fd[1], &c, 1);
	read(fd2[0], &c, 1);
	putchar('b');
	exit(0);
	}
	putchar('a');
	read(fd[0], &c, 1);
	putchar('c');
	write(fd2[1], &c, 1);
	wait(NULL);
	putchar('f');
	return 0;
	}	

3	Что будет выведено на экран? Если возможны несколько вариантов, привести все.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main(void)
	{
	char c = 'x';
	int fd[2], fd2[2];
	pipe(fd); pipe(fd2);
	if(fork() == 0) {
	write(fd[1], &c, 1);
		prxqs либо rpxqs либо rxpqs

№	Условие	Ответ

	write(1, "p", 1);
	read(fd2[0], &c, 1);
	write(1, "q", 1);
	exit(0);
	}
	write(1, "r", 1); read(fd[0], &c, 1);
	write(1, &c, 1); write(fd2[1], &c, 1);
	wait(NULL);
	write(1, "s", 1); return 0;
	}	

4	Что будет выведено на экран? Если возможны несколько вариантов– привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{ char c = 'x';
	int fd[2], fd2[2];
	pipe(fd); pipe(fd2);
	if(fork() == 0) {
	read(fd[0], &c, 1);
	putchar('d');
	read(fd2[0], &c, 1);
		acdbf
		либо
		adcbf

№	Условие	Ответ

	putchar('b');
	exit(0);
	}
	putchar('a');
	write(fd[1], &c, 1);
	putchar('c');
	write(fd2[1], &c, 1);
	wait(NULL);
	putchar('f');
	return 0;
	}	

5	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	char c = 'a';
	int fd[2], fd2[2];
	pipe(fd); pipe(fd2);
	if(fork() == 0) {
	write(fd[1], &c, 1);
		14235 либо 12435 либо 21435

№	Условие	Ответ

	putchar('2');
	read(fd2[0], &c, 1);
	putchar('3');
	exit(0);
	}
	putchar('1'); read(fd[0], &c, 1);
	putchar('4'); write(fd2[1], &c, 1);
	wait(NULL);
	putchar('5'); return 0;
	}	

6	Что будет выведено на экран? Если возможны несколько вариантов– привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	char c = 'b';
	int fd[2], fd2[2];
	pipe(fd); pipe(fd2);
	if(fork() == 0) {
	write(fd[1], &c, 1);
	putchar('e');
		acebf либо aecbf либо eacbf

№	Условие	Ответ

	read(fd2[0], &c, 1);
	putchar('b');
	exit(0);
	}
	putchar('a'); read(fd[0], &c, 1);
	putchar('c'); write(fd2[1], &c, 1);
	wait(NULL);
	putchar('f'); return 0;
	}	

7	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	char c = 'd';
	int fd[2], fd2[2];
	pipe(fd); pipe(fd2);
	if(fork() == 0) {
	write(fd[1], &c, 1);
	printf("3");
	read(fd2[0], &c, 1);
		02314 либо 03214 либо 30214

№	Условие	Ответ

	printf("1");;
	exit(0);
	}
	printf("0");;
	read(fd[0], &c, 1);
	printf("2");
	write(fd2[1], &c, 1);
	wait(NULL);
	printf("4");
	return 0;
	}	

8	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	char c = '1';
	int fd[2], fd2[2];
	pipe(fd); pipe(fd2);
	if(fork() == 0) {
	write(fd[1], &c, 1);
		45236
		либо
		24536
		либо
		42536

№	Условие	Ответ

	putchar('2');
	read(fd2[0], &c, 1);
	putchar('3');
	exit(0);
	}
	putchar('4');
	read(fd[0], &c, 1);
	putchar('5');
	write(fd2[1], &c, 1);
	wait(NULL);
	putchar('6');
	return 0;
	}	

9	Что будет выведено на экран? Если возможны несколько вариантов– привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main()
	{
	char c = 'a';
	int fd[2], fd2[2];
	pipe(fd); pipe(fd2);
		cdbf либо dcbf

№	Условие	Ответ

	if(fork() == 0) {
	write(fd[1], &c, 1);
	putchar('d');
	read(fd2[0], &c, 1);
	putchar('b'); exit(0);
	}
	read(fd[0], &c, 1); putchar('c');
	write(fd2[1], &c, 1); wait(NULL);
	putchar('f'); return 0;
	}	

Задача 23

№	Условие	Ответ
1	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. Вычеркните варианты вывода, невозможные при выполнении фрагментов программ тремя параллельными процессами (A,B,C): semId – идентификатор массива семафоров, состоящего из 1 семафора. Массив	Невозможные варианты вывода: 1) 12213 4) 11322

№	Условие	Ответ

	проинициализирован с помощью вызова:
	semctl(semId, 0, SETVAL, 2);
	Данный фрагмент выполняется двумя параллельными процессами (A и B):
	struct sembuf op; op.sem_num = 0; op.sem_flg = 0;
	op.sem_op = -1; semop(semId, &op, 1); putchar('1');
	putchar('2'); op.sem_op = 1; semop(semId, &op, 1);
	
	Данный фрагмент выполняется одним параллельным процессом C:
	struct sembuf op; op.sem_num = 0; op.sem_flg = 0;
	op.sem_op = -1; semop(semId, &op, 1); putchar('3');
	op.sem_op = 1; semop(semId, &op, 1);
	
	Варианты вывода:
	1) 12213
	2) 12123
	3) 11232
	4) 11322
	5) 11223
	6) 31212	

2	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.	Невозможные варианты вывода: 2) 23321 6) 22133

№	Условие	Ответ

	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	Вычеркните варианты вывода, невозможные при выполнении фрагментов программ тремя параллельными процессами (A,B,C):
	semId – идентификатор массива семафоров, состоящего из 1 семафора. Массив проинициализирован с помощью вызова:
	semctl(semId, 0, SETVAL, 2);
	Данный фрагмент выполняется двумя параллельными процессами (A и B):
	struct sembuf op; op.sem_num = 0; op.sem_flg = 0;
	op.sem_op = -1; semop(semId, &op, 1); putchar('2');
	putchar('3'); op.sem_op = 1; semop(semId, &op, 1);
	
	Данный фрагмент выполняется одним параллельным процессом C:
	struct sembuf op; op.sem_num = 0; op.sem_flg = 0;
	op.sem_op = -1; semop(semId, &op, 1); putchar('1');
	op.sem_op = 1; semop(semId, &op, 1);
	
	Варианты вывода:
	1) 23231
	2) 23321	

№	Условие	Ответ
	3) 22313 4) 22331 5) 12323 6) 22133 7) 21323	

3	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. Вычеркните варианты вывода, невозможные при выполнении фрагментов программ тремя параллельными процессами (X,Y,Z):
	semId – идентификатор массива семафоров, состоящего из 1 семафора. Массив проинициализирован с помощью вызова:
	semctl(semId, 0, SETVAL, 2);
	Данный фрагмент выполняется двумя параллельными процессами (X и Y):
	struct sembuf op; op.sem_num = 0; op.sem_flg = 0; op.sem_op = -1; semop(semId, &op, 1); putchar('a'); putchar('b'); op.sem_op = 1; semop(semId, &op, 1);
	
	Данный фрагмент выполняется одним параллельным процессом Z:
		Невозможные варианты вывода:
		1) abbac
		4) aacbb

№	Условие	Ответ

	struct sembuf op; op.sem_num = 0; op.sem_flg = 0;
	op.sem_op = -1; semop(semId, &op, 1); putchar('c');
	op.sem_op = 1; semop(semId, &op, 1);
	
	Варианты вывода:
	1) abbac
	2) ababc
	3) aabcb
	4) aacbb
	5) aabbc
	6) cabab	

4	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. Вычеркните варианты вывода, невозможные при выполнении фрагментов программ тремя параллельными процессами (A,B,C): semId – идентификатор массива семафоров, состоящего из 1 семафора. Массив проинициализирован с помощью вызова: semctl(semId, 0, SETVAL, 2); Данный фрагмент выполняется двумя параллельными процессами (A и B):	Невозможные варианты вывода: 1) 21123 4) 22311

№	Условие	Ответ

	struct sembuf op; op.sem_num = 0; op.sem_flg = 0;
	op.sem_op = -1; semop(semId, &op, 1); putchar('2');
	putchar('1'); op.sem_op = 1; semop(semId, &op, 1);
	
	Данный фрагмент выполняется одним параллельным процессом C:
	struct sembuf op; op.sem_num = 0; op.sem_flg = 0;
	op.sem_op = -1; semop(semId, &op, 1); putchar('3');
	op.sem_op = 1; semop(semId, &op, 1);
	
	Варианты вывода:
	1) 21123
	2) 21213
	3) 22131
	4) 22311
	5) 22113
	6) 32121	

5	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. Напишите варианты вывода (из указанных ниже), возможные при выполнении фрагментов программ тремя параллельными процессами (A,B,C):	Варианты вывода: tutuw ttuwu ttuuw wtutu

№	Условие	Ответ

	semId – идентификатор массива семафоров, состоящего из 1 семафора. Массив проинициализирован с помощью вызова:
	semctl(semId, 0, SETVAL, 2);
	Данный фрагмент выполняется двумя параллельными процессами (A и B):
	struct sembuf op;
	op.sem_num = 0;
	op.sem_flg = 0;
	op.sem_op = -1;
	semop(semId, &op, 1);
	putchar('t');
	putchar('u');
	op.sem_op = 1;
	semop(semId, &op, 1);
	
	Данный фрагмент выполняется одним параллельным процессом C:
	struct sembuf op;
	op.sem_num = 0;
	op.sem_flg = 0;
	op.sem_op = -1;
	semop(semId, &op, 1);
	putchar('w');	

№	Условие	Ответ

	op.sem_op = 1;
	semop(semId, &op, 1);
	
	Варианты вывода:
	1) tuutw
	2) tutuw
	3) ttuwu
	4) ttwuu
	5) ttuuw
	6) wtutu	

Задача 24

№	Условие	Ответ
1	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. semId – идентификатор массива семафоров, состоящего из 1 семафора Массив проинициализирован с помощью вызова: semctl(semId, 0, SETVAL, 5);	Варианты: 121212 121122 112122 112212

№	Условие	Ответ

	Фрагмент программы выполняется 3-мя параллельными процессами:
	struct sembuf op; op.sem_num = 0; op.sem_flg = 0;
	op.sem_op = -2; semop(semId, &op, 1);
	write(1, "1", 1); write(1, "2", 1);
	op.sem_op = 2; semop(semId, &op, 1);	

2	Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	semId – идентификатор массива семафоров, состоящего из 1 семафора
	Массив проинициализирован с помощью вызова:
	semctl(semId, 0, SETVAL, 8);
	Фрагмент программы выполняется 3-мя параллельными процессами:
	struct sembuf op; op.sem_num = 0; op.sem_flg = 0;
	op.sem_op = -3; semop(semId, &op, 1);
	write(1, "1", 1); write(1, "2", 1);
	op.sem_op = 2; semop(semId, &op, 1);
		Варианты:
		121212
		121122
		112122
		112212

№	Условие	Ответ

3	Перечислите все варианты вывода этой программы.
	Операции с семафорами условно названы как down и up.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main(void)
	{
	semaphore s = 0;
	if (fork() == 0) {
	down(&s);
	up(&s);
	write(1, “a”, 1);
	up(&s);
	} else {
	write(1, “b”, 1);
	up(&s);
	write(1, “c”, 1);
	}
	return 0;
	}
		bca
		bac

4	Перечислите все варианты вывода этой программы. Операции с семафорами условно названы как down и up.	bac abc bca

№	Условие	Ответ

	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	int main(void)
	{
	semaphore s = 1;
	if (fork() == 0) {
	down(&s);
	write(1, “a”, 1);
	up(&s);
	} else {
	write(1, “b”, 1);
	down(&s);
	up(&s);
	write(1, “c”, 1);
	}
	return 0;
	}	

5	Какие средства синхронизации можно выбрать и какие действия вставить в эту программу, чтобы сначала напечаталось AB или BA (оба варианта должны быть возможны), потом CD?
	int main(void) {
	if (fork()) {
	write(1, “A”, 1);
		Можно использовать семафор:
		int main(void) {
		semaphore s = 0;
		if (fork()) {
		write(1, “A”, 1);
		up(&s, 1);
		down(&s, 2);

№	Условие	Ответ

	write(1, “C”, 1);
	wait(NULL);
	} else {
	write(1, “B”, 1);
	write(1, “D”, 1);
	}
	}
		write(1, “C”, 1);
		up(&s, 3);
		wait(NULL);
		} else {
		write(1, “B”, 1);
		up(&s, 1);
		down(&s, 3);
		write(1, “D”, 1);
		}
		}

6	Какие средства синхронизации можно выбрать и какие действия вставить в эту программу, чтобы сначала напечаталось AB, потом CD или DC (оба варианта должны быть возможны)?
	int main(void) {
	if (fork()) {
	write(1, “A”, 1);
	write(1, “C”, 1);
	wait(NULL);
	} else {
	write(1, “B”, 1);
	write(1, “D”, 1);
	}
	}
		Можно использовать семафор:
		int main(void) {
		semaphore s = 0;
		if (fork()) {
		write(1, “A”, 1);
		up(&s, 1);
		down(&s, 2);
		write(1, “C”, 1);
		wait(NULL);
		} else {
		down(&s, 1);
		write(1, “B”, 1);
		up(&s, 2);
		write(1, “D”, 1);

№	Условие	Ответ
		} }

7	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. Напишите все варианты вывода при выполнении фрагментов программ тремя параллельными процессами (A, B, C):
	semId – идентификатор массива семафоров, состоящего из 1 семафора. Массив проинициализирован с помощью вызова:
	semctl(semId, 0, SETVAL, 1);
	Данный фрагмент выполняется двумя параллельными процессами (A и B):
	struct sembuf op; op.sem_num = 0; op.sem_flg = 0; op.sem_op = -1; semop(semId, &op, 1); putchar('1'); putchar('2'); op.sem_op = 1; semop(semId, &op, 1);
	
	Данный фрагмент выполняется одним параллельным процессом C:
	struct sembuf op; op.sem_num = 0; op.sem_flg = 0; op.sem_op = -1; semop(semId, &op, 1); putchar('3'); op.sem_op = 1;
		12123
		12312
		31212

№	Условие	Ответ
	semop(semId, &op, 1);	

8	Опишите словесно все варианты вывода на экран.
	
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	semId – идентификатор массива семафоров, состоящего из 1 семафора
	Массив проинициализирован с помощью вызова:
	semctl(semId, 0, SETVAL, 3);
	Фрагмент программы выполняется миллионом параллельных процессов:
	struct sembuf op; op.sem_num = 0; op.sem_flg = 0;
	op.sem_op = -2;
	semop(semId, &op, 1);
	write(1,"a",1); write(1,"b",1);
	op.sem_op = 2;
	semop(semId, &op, 1);
		Будет выведена последовательность из миллиона пар ab

9	Опишите словесно все варианты вывода на экран Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.	Будет выведена последовательность из a и b длиной в 32 символа, в которой содержится по 16 символов a и b. Каждый префикс этой последовательности содержит количество символов a, которое больше либо равно количества

№	Условие	Ответ

	int main()
	{
	int fd[2];
	char c[2];
	pipe(fd);
	write(fd[1], c, 2);
	fork();fork();fork();fork();
	read(fd[0], c, 1);
	write(1, "a", 1);
	write(1, "b", 1);
	write(fd[1], c, 1);
	wait(NULL);
	return 0;
	}
		символов b в этом префиксе. Возможен любой вариант такой последовательности

10	Опишите словесно все варианты вывода на экран. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. semId – идентификатор массива семафоров, состоящего из 1 семафора Массив проинициализирован с помощью вызова: semctl(semId, 0, SETVAL, 20);	Будет выведена последовательность из 1 и 2 длиной в 40 символов, в которой содержится по 20 единиц и 20 двоек. Каждый префикс этой последовательности содержит количество единиц, которое больше либо равно количества двоек в этом префиксе. Возможен любой вариант такой последовательности.

№	Условие	Ответ

	Фрагмент программы выполняется двадцатью параллельными процессами:
	struct sembuf op; op.sem_num = 0; op.sem_flg = 0;
	op.sem_op = -1;
	semop(semId, &op, 1);
	write(1,"1",1); write(1,"2",1);
	op.sem_op = 1;
	semop(semId, &op, 1);	

11	Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	semId – идентификатор массива семафоров, состоящего из 1 семафора
	Массив проинициализирован с помощью вызова:
	semctl(semId, 0, SETVAL, 5);
	Фрагмент программы выполняется 3-мя параллельными процессами:
	struct sembuf op; op.sem_num = 0; op.sem_flg = 0;
	op.sem_op = -3; semop(semId, &op, 1);
	write(1, "1", 1); write(1, "2", 1);
	op.sem_op = 2; semop(semId, &op, 1);
		Варианты:
		121212

№	Условие	Ответ

12	Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	semId – идентификатор массива семафоров, состоящего из 1 семафора
	Массив проинициализирован с помощью вызова:
	semctl(semId, 0, SETVAL, 5);
	Фрагмент программы выполняется 3-мя параллельными процессами:
	struct sembuf op; op.sem_num = 0; op.sem_flg = 0;
	op.sem_op = -2; semop(semId, &op, 1);
	write(1, "a", 1); write(1, «b", 1);
	op.sem_op = 2; semop(semId, &op, 1);
		Варианты:
		ababab
		abaabb
		aababb
		aabbab

13	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. semId – идентификатор массива семафоров, состоящего из 1 семафора	Варианты: efefef efeeff eefeff eeffef

№	Условие	Ответ

	Массив проинициализирован с помощью вызова:
	semctl(semId, 0, SETVAL, 7);
	Фрагмент программы выполняется 3-мя параллельными процессами:
	struct sembuf op;
	op.sem_num = 0;
	op.sem_flg = 0;
	op.sem_op = -3;
	semop(semId, &op, 1);
	putchar('e');
	putchar('f');
	op.sem_op = 3;
	semop(semId, &op, 1);	

14	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. Перечислите возможные варианты вывода при выполнении фрагментов программ тремя параллельными процессами (A, B, C): semId – идентификатор массива семафоров, состоящего из 1 семафора. Массив проинициализирован с помощью вызова: semctl(semId, 0, SETVAL, 1);	XXYZ YZXX XYZX

№	Условие	Ответ

	Данный фрагмент выполняется двумя параллельными процессами (A и B):
	struct sembuf op; op.sem_num = 0; op.sem_flg = 0;
	op.sem_op = -1; semop(semId, &op, 1);
	putchar('X');
	op.sem_op = 1; semop(semId, &op, 1);
	
	Данный фрагмент выполняется одним параллельным процессом C:
	struct sembuf op; op.sem_num = 0; op.sem_flg = 0;
	op.sem_op = -1; semop(semId, &op, 1);
	putchar('Y');
	putchar('Z');
	op.sem_op = 1; semop(semId, &op, 1);	

Задача 25

№	Условие	Ответ
1	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.	Варианты: ababab abaabb aababb aabbab

№	Условие	Ответ

	int main()
	{
	int fd[2];
	char c[2] ="34";
	pipe(fd);
	write(fd[1], c, 2);
	if(fork())
	fork();
	read(fd[0], c, 1);
	write(1, "a", 1);
	write(1, "b", 1);
	write(fd[1], c, 1);
	wait(NULL);
	return 0;
	}	

2	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. int main()	Варианты: 121212 121122 112122 112212

№	Условие	Ответ

	{
	int fd[2];
	char c[2] ="34";
	pipe(fd);
	write(fd[1], c, 2);
	if(fork())
	fork();
	read(fd[0], c, 1);
	write(1, "1", 1);
	write(1, "2", 1);
	write(fd[1], c, 1);
	wait(NULL);
	return 0;
	}	

3	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. int main() { int fd[2];	Будет выведена последовательность пар ab длиной 12 (всего 24 символа)

№	Условие	Ответ

	char c[2] ="34";
	pipe(fd);
	write(fd[1], c, 2);
	if (fork())
	fork();
	fork(); fork();
	read(fd[0], c, 2);
	write(1, "a", 1);
	write(1, "b", 1);
	write(fd[1], c, 2);
	wait(NULL);
	return 0;
	}	

4	Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. int main() { int fd[2]; char d[2] ="34";	Варианты: хухуху хуххуу ххухуу ххууху

№	Условие	Ответ

	pipe(fd);
	write(fd[1], d, 2);
	if(fork())
	fork();
	read(fd[0], d, 1);
	write(1, "x", 1);
	write(1, "y", 1);
	write(fd[1], d, 1);
	wait(NULL);
	return 0;
	}	

5	Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	
	int main()
	{
	int fd[2];
	char c[2]="34";
	pipe(fd);
	write(fd[1], c, 2);
		Варианты:
		343434
		343344
		334344
		334434

№	Условие	Ответ

	if(fork())
	fork();
	read(fd[0], c, 1);
	putchar('3');
	putchar('4');
	write(fd[1], c, 1);
	wait(NULL);
	return 0;
	}	

Задача 26

№	Условие	Ответ

1	Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	
	int main()
	{
	int fd[2];
	pipe(fd);
		a
		ba
		ba
		
		либо
		
		a
		ab
		ba

№	Условие	Ответ

	char x[] = "ba\n";
	if(fork()) { puts(x + 1); write(fd[1], x, 1);
	wait(NULL);
	}
	else { write(fd[1], &x[1], 1); read(fd[0], x, 1);
	read(fd[0], x+1, 1);
	}
	puts(x); return 0;
	}	

2	Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	
	int main()
	{
	int fd[2];
	pipe(fd);
	char x[] = "ba\n";
	if(fork()) { write(fd[1], x, 1); wait(NULL);
	}
		ba
		ba
		
		либо
		
		ab
		ba

№	Условие	Ответ

	else { write(fd[1], &x[1], 1); read(fd[0], x, 1);
	read(fd[0], x+1, 1);
	}
	puts(x); return 0;
	}	

3	Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	
	int main()
	{
	int fd[2];
	pipe(fd);
	char x[] = "dc\n";
	if(fork()) { puts(x + 1); write(fd[1], x, 1);
	wait(NULL);
	}
	else { write(fd[1], &x[1], 1); read(fd[0], x, 1);
	read(fd[0], x+1, 1);
	}
	puts(x); return 0;
		c
		dc
		dc
		
		либо
		
		c
		cd
		dc

№	Условие	Ответ
	}	

4	Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	
	int main()
	{
	int fd[2];
	pipe(fd);
	char x[] = "97\n";
	if(fork()) {
	puts(x + 1);
	write(fd[1], x, 1);
	wait(NULL);
	}
	else {
	write(fd[1], &x[1], 1);
	read(fd[0], x, 1);
	read(fd[0], x+1, 1);
	}
	puts(x);
		7
		97
		97
		
		либо
		
		7
		79
		97

№	Условие	Ответ
	return 0; }	

Задача 27

№	Условие	Ответ

1	Описать, что произойдет с процессами и почему (все системные вызовы отрабатывают корректно – без отказов)
	
	int main(int argc, char **argv)
	{
	int fd[2];
	char c[2] = "ab";
	pipe(fd);
	if(fork()) { /* процесс №1 */
	close (fd[0]);
	close (fd[1]);
	} else { /* процесс №2 */
	read(fd[0], c,1);
	}
	}
		Процесс № 1: завершится.
		процесс № 2: зависнет, т.к. в системе не будет закрыт дескриптор записи в этот канал (дескриптор, который был унаследован процессом №2).

№	Условие	Ответ

2	Описать, что произойдет с процессами и почему (все системные вызовы отрабатывают корректно – без отказов)
	
	int main(int argc, char **argv)
	{
	int fd[2];
	char c[2] = {};
	pipe(fd);
	if(fork()) { /* процесс №1 */
	close (fd[0]);
	close (fd[1]);
	write(fd[1], c,1);
	} else { /* процесс №2 */
	read(fd[0], c,1);
	}
	return 0;
	}
		Процесс № 1: завершится с кодом 0.
		процесс № 2: зависнет, т.к. в системе не будет закрыт дескриптор записи в этот канал (дескриптор, который был унаследован процессом №2).

3	Описать, что произойдет с процессами и почему (все системные вызовы отрабатывают корректно – без отказов) int main(void) {	Поведение не определено, т.к. не инициализирован массив fd.

№	Условие	Ответ

	int fd[2];
	char c[] = "abc\n";
	if(fork() == 0) { /* процесс №1 */
	close (fd[0]);
	close (fd[1]);
	} else { /* процесс №2 */
	read(fd[0], &c, 2);
	}
	}	

4	Описать, что произойдет с процессами и почему (все системные вызовы отрабатывают корректно – без отказов)
	
	int main(int argc, char **argv)
	{
	int fd[2];
	char c[2] = "ab";
	pipe(fd);
	if(fork()) { /* процесс №1 */
	close (fd[0]);
	close (fd[1]);
	} else { /* процесс №2 */
	read(fd[0], c,1);
	close (fd[1]);
		процесс № 1: завершится.
		процесс № 2: зависнет, т.к. в системе не будет закрыт дескриптор записи в этот канал (дескриптор, который был унаследован процессом №2).

№	Условие	Ответ
	} }	

5	Описать, что произойдет с процессами и почему (все системные вызовы отрабатывают корректно – без отказов)
	
	int main(int argc, char **argv)
	{
	int fd[2];
	char c[2] = "ab";
	pipe(fd);
	if(fork()) {/*процесс №1*/
	close (fd[0]);
	close (fd[1]);
	wait(NULL);
	} else {/*процесс №2*/
	read(fd[0],c,1);
	close(fd[1]);
	}
	}
		процесс № 2: зависнет, т.к. в системе не будет закрыт дескриптор записи в этот канал (дескриптор, который был унаследован процессом №2).
		процесс № 1: зависнет в ожидании завершения процесса № 2.

6	Описать, что произойдет с процессами и почему (все системные вызовы отрабатывают корректно – без отказов)	процесс № 1: заблокируется. процесс № 2: зависнет, т.к. в системе не будет закрыт дескриптор записи в этот канал

№	Условие	Ответ

	int main(int argc, char **argv)
	{
	int fd[2];
	char c[2] = "ab";
	pipe(fd);
	if(fork()) { /* процесс №1 */
	wait(NULL);
	close(fd[0]);
	close(fd[1]);
	} else { /* процесс №2 */
	read(fd[0], c,1);
	}
	}
		(дескриптор, который был унаследован процессом №2).

7	Описать, что произойдет с процессами и почему (все системные вызовы отрабатывают корректно – без отказов)
	
	int main(int argc, char **argv)
	{
	int fd[2];
	char d[2] = "ab";
	pipe(fd);
	if(fork()) { /* процесс №1 */
	read(fd[0], d,1);
		процесс № 2: завершится.
		процесс № 1: зависнет, т.к. в системе не будет закрыт дескриптор записи в этот канал

№	Условие	Ответ

	read(fd[0],d, 2);
	close(fd[1]);
	} else { /* процесс №2 */
	write(fd[1], d, 1);
	close(fd[0]);
	close(fd[1]);
	}
	}	

8	Описать, что произойдет с процессами и почему (все системные вызовы отрабатывают корректно – без отказов)
	int
	main(int argc, char **argv)
	{
	int buf = 1, fd[2]; pipe(fd);
	if (fork()) { /* процесс №1 */
	write (fd[1], &buf, sizeof(int));
	} else {/* процесс №2 */
	while (read(fd[0], &buf, sizeof(int))) {
	buf++;
	};
	printf("%d\n", buf);
	}
	return 0;
		Процесс №1 выполнит действия и завершится. Процесс №2 считает информацию из канала и зависнет (в нем не закрыт «пишуший» дескриптор канала и read будет ожидать его закрытия).

№	Условие	Ответ
	}	

9	Описать, что произойдет с процессами и почему (все системные вызовы отрабатывают корректно – без отказов)
	
	int main(int argc, char **argv)
	{
	int fd[2];
	char c[2] = "cd";
	pipe(fd);
	if(fork()) { /* процесс №1 */
	close (fd[0]);
	close (fd[1]);
	wait(NULL);
	} else { /* процесс №2 */
	read(fd[0], c,1);
	_exit(0);
	}
	exit(0);
	}
		процесс № 2: зависнет, т.к. в системе не будет закрыт дескриптор записи в этот канал (дескриптор, который был унаследован процессом №2).
		процесс № 1: зависнет на ожидании процесса №2.

10	Описать, что произойдет с процессами и почему (все системные вызовы отрабатывают корректно – без отказов)	процесс № 2: зависнет, т.к. в системе не будет закрыт дескриптор записи в этот канал (дескриптор, который остался у родительского процесса №1).

№	Условие	Ответ

	int main(int argc, char **argv)
	{
	int fd[2];
	char c[2] = "ab";
	pipe(fd);
	if(!fork()) { /* процесс №2 */
	close (fd[1]);
	read(fd[0], c,1);
	_exit(0);
	}
	/* процесс №1 */
	close (fd[0]);
	wait(NULL);
	exit(0);
	}
		процесс № 1: зависнет на ожидании процесса №2.

Задача 28

№	Условие	Ответ
1	Содержимое файла “1.txt” – строка «abcde». Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.	Ответ: aad либо cad Комментарий для проверяющего: Два варианта порождаются за счет «гонок» между lseek и read в «сыне». Сам вывод определяется тем, что при «наследовании» и дублировании

№	Условие	Ответ

	Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	
	int main()
	{
	  char c = 'a';
	  int fd;
	
	  fd = open("1.txt", O_RDONLY);
	
	  if(fork())
	  {
	  int fd2 = open("1.txt", O_RDONLY);
	  int fd3 = dup(fd);
	  lseek(fd, 2, SEEK_CUR);
	  wait(NULL);
	  read(fd2, &c, 1); write(1, &c, 1);
	  read(fd3, &c, 1); write(1, &c, 1);
	  }
	  else
	  { read(fd, &c, 1); write(1, &c, 1); }
	
	 return 0;
	}
		файлового дескриптора файловый указатель является общим, а при open создается новый.

№	Условие	Ответ

2	Содержимое файла “1.txt” – строка «examos». Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	
	int main()
	{
	  char c = 'a';
	  int fd;
	
	  fd = open("1.txt", O_RDONLY);
	
	  if(fork())
	  {
	  int fd2 = open("1.txt", O_RDONLY);
	  int fd3 = dup(fd);
	  lseek(fd, 2, SEEK_CUR);
	  wait(NULL);
	  read(fd2, &c, 1); write(1, &c, 1);
	  read(fd3, &c, 1); write(1, &c, 1);
	  }
		Ответ: eem либо aem
		Комментарий для проверяющего:
		Два варианта порождаются за счет «гонок» между lseek и read в «сыне». Сам вывод определяется тем, что при «наследовании» и дублировании файлового дескриптора файловый указатель является общим, а при open создается новый.

№	Условие	Ответ

	else
	{ read(fd, &c, 1); write(1, &c, 1); }
	
	return 0;
	}	

3	Содержимое файла “1.txt” – строка «123456». Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	
	int main(void)
	{
	char c = '0';
	int fd = open("1.txt", O_RDONLY);
	if(fork() == 0) {
	read(fd, &c, 1); write(1, &c, 1);
	} else {
	int fd2 = open("1.txt", O_RDONLY);
	int fd3 = dup(fd);
	lseek(fd, 2, SEEK_CUR);
	wait(NULL);
	read(fd2, &c, 1); write(1, &c, 1);
	read(fd3, &c, 1); write(1, &c, 1);
		Комментарий для проверяющего:
		Два варианта порождаются за счет «гонок» между lseek и read в «сыне». Сам вывод определяется тем, что при «наследовании» и дублировании файлового дескриптора файловый указатель является общим, а при open создается новый.
		Ответ:
		314 (lseek->read)
		114 (read -> lseek)

№	Условие	Ответ
	} return 0; }	

4	Содержимое файла “1.txt” – строка «abcdef». Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	
	int main()
	{
	char c = 'a';
	int fd;
	fd = open("1.txt", O_RDONLY);
	
	if(fork())
	{
	int fd2 = open("1.txt",O_RDONLY);
	int fd3 = dup(fd);
	lseek(fd, -2, SEEK_END);
	wait(NULL);
	read(fd2, &c, 1);
		Ответ: aae либо eaf
		Комментарий для проверяющего:
		Два варианта порождаются за счет «гонок» между lseek и read в «сыне». Сам вывод определяется тем, что при «наследовании» и дублировании файлового дескриптора файловый указатель является общим, а при open создается новый.

№	Условие	Ответ

	write(1, &c, 1);
	read(fd3, &c, 1);
	write(1, &c, 1);
	} else {
	read(fd, &c, 1);
	write(1, &c, 1);
	}
	return 0;
	}	

5	Содержимое файла “1.txt” – строка «abcde». Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	
	int main()
	{
	char c = 'a';
	int fd;
	fd = open("1.txt", O_RDONLY);
	
	if(fork())
		Ответ: aac либо cad
		Комментарий для проверяющего:
		Два варианта порождаются за счет «гонок» между lseek и read в «сыне». Сам вывод определяется тем, что при «наследовании» и дублировании файлового дескриптора файловый указатель является общим, а при open создается новый.

№	Условие	Ответ

	{
	int fd2 = open("1.txt",O_RDONLY);
	int fd3 = dup(fd);
	lseek(fd, 2, SEEK_SET);
	wait(NULL);
	read(fd2, &c, 1);
	write(1, &c, 1);
	read(fd3, &c, 1);
	write(1, &c, 1);
	} else {
	read(fd, &c, 1);
	write(1, &c, 1);
	}
	return 0;
	}	

6	Содержимое файла “1.txt” – строка «edcba». Что будет выведено на экран? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. int main()	Ответ: eeb либо ceb Комментарий для проверяющего: Два варианта порождаются за счет «гонок» между lseek и read в «сыне». Сам вывод определяется тем, что при «наследовании» и дублировании файлового дескриптора файловый указатель является общим, а при open создается новый.

№	Условие	Ответ

	{
	char c = 'a';
	int fd;
	
	fd = open("1.txt", O_RDONLY);
	
	if(fork())
	{
	int fd2 = open("1.txt", O_RDONLY);
	int fd3 = dup(fd);
	lseek(fd, 2, SEEK_CUR);
	wait(NULL);
	read(fd2, &c, 1); write(1, &c, 1);
	read(fd3, &c, 1); write(1, &c, 1);
	}
	else
	{ read(fd, &c, 1); write(1, &c, 1); }
	
	return 0;
	}	

7	Содержимое файла b.txt – строка "12345". Что будет выведено на экран при выполнении этого фрагмента программы? Если допустимы несколько вариантов вывода приведите их все. Варианты	114 либо 314

№	Условие	Ответ

	разделить словом «либо». Считать, что все системные вызовы выполнены успешно, обращение к функции вывода работает атомарно и без буферизации. Подключение заголовочных файлов опущено.
	int main()
	{
	char c;
	int fd;
	
	fd = open("b.txt", O_RDONLY);
	
	if (fork())
	{
	int fd2 = open("b.txt", O_RDONLY);
	lseek(fd,2,SEEK_CUR);
	wait(NULL);
	read(fd2, &c, 1); write(1, &c, 1);
	read(fd, &c, 1); write(1, &c, 1);
	}
	else
	{
	read(fd, &c, 1); write(1, &c, 1);
	}
	return 0;	

№	Условие	Ответ
	}	

8	Описать, что произойдет с процессами и почему (все системные вызовы отрабатывают атомарно и корректно – без отказов). Что будет выведено на экран? Если допустимы несколько вариантов вывода, приведите все.
	int main(int argc, char **argv)
	{
	int fd[2], buf = 10;
	pipe(fd);
	if(!fork()) {
	write (fd[1], &buf, sizeof(int)); buf++;
	printf("%d\n", buf);
	} else {
	while(read(fd[0], &buf, sizeof(int))) {
	buf--; printf("%d\n", buf);
	}
	}
	return 0;
	}
		1. Сыновий процесс выполнит действия и завершится.
		2. Родительский процесс считает информацию из канала, выведет на экран «9» и зависнет (в нем не закрыт «пишуший» дескриптор канала и read будет ожидать его закрытия).
		
		На экран будет выведено:
		11
		9
		ИЛИ
		9
		11

9	Содержимое файла “1.txt” – строка «HelloWorld». Что будет выведено на экран? Если возможны несколько вариантов – привести все.	WoHr или HoHr Комментарий для проверяющего:

№	Условие	Ответ

	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.
	
	int main()
	{
	  char c = 'a';
	  int fd;
	
	  fd = open("1.txt", O_RDONLY);
	
	  if(fork())
	  {
	  int fd2 = open("1.txt", O_RDONLY);
	  int fd3 = dup(fd);
	  lseek(fd, 5, SEEK_CUR);
	  wait(NULL);
	  read(fd, &c, 1); write(1, &c, 1);
	  read(fd2, &c, 1); write(1, &c, 1);
	  read(fd3, &c, 1); write(1, &c, 1);
	  }
	  else
	  {
		Два варианта порождаются за счет «гонок» между lseek и read в «сыне». Сам вывод определяется тем, что при «наследовании» и дублировании файлового дескриптора файловый указатель является общим, а при open создается новый.

№	Условие	Ответ

	read(fd, &c, 1); write(1, &c, 1);
	}
	
	return 0;
	}	

10	Содержимое файла “1.txt” – строка «HelloWorld». Что будет выведено на экран? Если возможны несколько вариантов – привести все.
	Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации.
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	
	int main()
	{
	char c = 'a';
	int fd;
	
	fd = open("1.txt", O_RDONLY);
	
	if(fork())
	{
	int fd2 = open("1.txt", O_RDONLY);
	int fd3 = dup(fd);
		oWHo или HoHW
		
		Комментарий для проверяющего:
		Два варианта порождаются за счет «гонок» между lseek и read в «сыне». Сам вывод определяется тем, что при «наследовании» и дублировании файлового дескриптора файловый указатель является общим, а при open создается новый.

№	Условие	Ответ

	lseek(fd, -6, SEEK_END);
	wait(NULL);
	read(fd3, &c, 1); write(1, &c, 1);
	read(fd2, &c, 1); write(1, &c, 1);
	read(fd, &c, 1); write(1, &c, 1);
	}
	else
	{
	read(fd, &c, 1); write(1, &c, 1);
	}
	
	return 0;
	}	

11	Содержимое файла “1.txt” – строка «abcde». Возможны ли варианты вывода: cad и abc? Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено. int main() { char c = 'a';	Возможны: aad либо cad Невозможны: abc Комментарий для проверяющего: Два варианта порождаются за счет «гонок» между lseek и read в «сыне». Сам вывод определяется тем, что при «наследовании» и дублировании файлового дескриптора файловый указатель является общим, а при open создается новый.

№	Условие	Ответ

	int fd;
	
	fd = open("1.txt", O_RDONLY);
	
	if(fork())
	{
	int fd2 = open("1.txt", O_RDONLY);
	int fd3 = dup(fd);
	lseek(fd, 2, SEEK_CUR);
	wait(NULL);
	read(fd2, &c, 1); write(1, &c, 1);
	read(fd3, &c, 1); write(1, &c, 1);
	}
	else
	{ read(fd, &c, 1); write(1, &c, 1); }
	
	return 0;
	}	

Задача 29

№	Условие	Ответ
1	В файловой системе используются битовые массивы для хранения информации о свободных и занятых	// Предположим, CHAR_BIT == 8, блоки пронумерованы от 0 до Max_Num

№	Условие	Ответ

	блоках. Написать на Си функцию, принимающую в качестве параметров указатель на начало этого битового массива (последовательность байтов), номер блока файловой системы, максимально возможный номер блока и возвращающую статус занятости этого блока: 0 – свободен, 1 занят, -1 номер вне диапазона.	int is_free(unsigned char *BitBlocks, unsigned Num, unsigned Max_Num) {
		if (Num >= Max_Num) {
		return -1;
		} else {
		return (BitBlocks[Num >> 3] >> (7 - (Num & 7u))) & 1;
		}
		}

2	Головка HDD находится на дорожке 100. Нужно выполнить следующие запросы к дорожкам: 60, 5, 22, 83, 120, 71. Назвать последовательность запросов при использовании жадного алгоритма.	На каждом шаге выбираем ближайшую дорожку. Получаем: 83, 71, 60, 22, 5, 120.
3	Головка HDD находится на дорожке 90. Нужно выполнить следующие запросы к дорожкам: 104, 20, 95, 56, 81, 3. Назвать последовательность запросов при использовании жадного алгоритма.	На каждом шаге выбираем ближайшую дорожку. Получаем: 95, 104, 81, 56, 20, 3.
4	Перечислите все ситуации, в которых системный вызов open(“/home/ira/dir/file”, O_RDONLY) может вернуть -1.	Замечание: не надо требовать перечислить ВСЕ ошибки, возникающие в ходе выполнения open, их в man – около 20. Достаточно назвать 3-4 различных ситуации, например: - файла /home/ira/dir/file нет

№	Условие	Ответ
		- какая-то из директорий в пути /home/ira/dir/file отсутствует или недоступна - пользователь не имеет прав на чтение файла /home/ira/dir/file - в процессе нет доступных файловых дескрипторов для открытия - превышен системный лимит на число открытых файлов - файл /home/ira/dir/file заблокирован для чтения др.
5	Перечислите все ситуации, в которых системный вызов execl("/home/igor/dir/prog") может вернуть -1.	Замечание: не надо требовать перечислить ВСЕ ошибки, возникающие в ходе выполнения exec, их в man – около 20. Достаточно назвать 3-4 различных ситуации, например: - файла /home/igor/dir/prog нет - какая-то из директорий в пути /home/igor/dir/ отсутствует или недоступна - пользователь не имеет прав на выполнение файла /home/igor/dir/prog - файл /home/igor/dir/prog не является исполняемым - файл /home/igor/dir/prog является исполняемым, но имеет неверный формат - в процессе нет доступных файловых дескрипторов для открытия

№	Условие	Ответ
		- нет ресурсов ядра для загрузки нового тела процесса - файл /home/igor/dir/prog заблокирован для чтения др.

6	Пусть в некоторой ОС используется файловая система, использующая FAT. Для представления номера блока в системе используется беззнаковое целое. Написать функцию, которая по номеру начального блока файла определяет размер файла в блоках. Функция принимает в качестве параметров номер начального блока файла и указатель на область памяти, в которой находится FAT.	i-ая строка таблицы FAT хранит информацию о состоянии i-ого блока файловой системы, а, кроме того, в ней указывается номер следующего блока файла. Для получения списка блоков файловой системы, в которых хранится содержимое конкретного файла, необходимо найти номер начального блока, а затем, последовательно обращаясь к таблице размещения и извлекая из каждой записи номер следующего блока, дойти до ссылки на нулевую строку таблицы. Нулевая строка таблицы уже не относится к рассматриваемому файлу.
		
		int
		calculateSize(unsigned int num, unsigned int *fat)
		{
		int counter = 0;
		while (num != 0) {
		num = fat[num];
		counter++;

№	Условие	Ответ
		} return counter; }

Задача 30

№	Условие	Ответ
1	Сколько раз система обратится к содержимому индексных дескрипторов при вызове: open("/dir/dir/file", O_RDONLY) ? Прокомментировать, почему? Считаем, что ни один из элементов пути к файлу не является символической ссылкой.	4 раза. 1 - дескриптор для / - чтобы найти дескриптор для файла каталога dir 2 - дескриптор для /dir - чтобы найти дескриптор для файла каталога /dir/dir 4 - дескриптор для /dir/dir/ - чтобы найти дескриптор для файла /dir/dir/ file 5 - дескриптор для /dir/dir/file – чтобы проверить права доступа для этого файла и последующего чтения в память.
2	Сколько раз система обратится к содержимому индексных дескрипторов при вызове: open("/dir1/dir2/dir3/file", O_RDONLY)? Прокомментировать, почему? Считаем, что ни один из элементов пути к файлу не является символической ссылкой.	5 раз. 1 - дескриптор для / - чтобы найти дескриптор для файла каталога dir1 2 - дескриптор для /dir1 - чтобы найти дескриптор для файла каталога /dir1/dir2 3 - дескриптор для /dir1/dir2 - чтобы найти дескриптор для файла каталога /dir1/dir2/dir3

№	Условие	Ответ
		4 - дескриптор для /dir1/dir2/dir3 - чтобы найти дескриптор для файла /dir1/dir2/dir3/file 5 - дескриптор для /dir1/dir2/dir3/file – чтобы проверить права доступа для этого файла и последующего чтения в память.
3	Сколько раз система обратится к содержимому индексных дескрипторов при вызове: open("/dir1/file", O_RDONLY) ? Прокомментировать, почему? Считаем, что ни один из элементов пути к файлу не является символической ссылкой.	3 раза. 1 - дескриптор для / - чтобы найти дескриптор для файла каталога dir1 2 - дескриптор для /dir1 - чтобы найти дескриптор для файла /dir1/file 3 - дескриптор для /dir1/file – чтобы проверить права доступа для этого файла и последующего чтения в память.
4	Сколько индексных дескрипторов нужно прочитать, чтобы загрузить файл /usr/exm/file.dat ? Прокомментировать, почему? Считаем, что ни один из элементов пути к файлу не является символической ссылкой.	4
5	Сколько индексных дескрипторов нужно прочитать, чтобы загрузить файл /home/program/dz/files/task.c ? Прокомментировать, почему? Считаем, что ни один из элементов пути к файлу не является символической ссылкой.	6

Дополнительные задачи

№	Условие	Ответ

1	Каким будет содержание файла file после каждого из вызовов write?
	Все системные вызовы прорабатывают успешно.
	Подключение заголовочных файлов опущено.
	
	fd1 = open(file, O_RDWR | O_CREAT | O_TRUNC, S_IRUSR | S_IWUSR);
	fd2 = dup(fd1);
	fd3 = open(file, O_RDWR);
	write(fd1, "Hello,", 6); //1
	write(fd2, "world", 6); //2
	lseek(fd2, 0, SEEK_SET);
	if(!fork)
	{
	write(fd1, "HELLO,", 6); //3
	write(fd3, "123456", 6); //4
	_exit(0);
	}
	wait(NULL);
	write(fd2, "ByeBye", 6); //5
		1 – Hello,
		2 – Hello,world
		3 – HELLO,world
		4 – 123456world
		5 – 123456ByeBye

№	Условие	Ответ

2	Дан код программы. Подключаемые файлы опущены. Считая, что все вызовы отработают успешно, написать, что будет выведено в файл "a.txt"
	int main()
	{
	int fd;
	char *str="abcdefgh";
	char *dig="1234567";
	fd = creat("a.txt", 0644);
	write(fd, str, 2);
	write(fd, str+3, 3);
	lseek(fd, 0, SEEK_SET);
	write(fd, dig, 1);
	lseek(fd, 0, SEEK_END);
	write(fd, dig+2, 2);
	return 0;
	}
		1bdef34

3	Дан фрагмент программы. Что будет выведено на экран при выполнении этого фрагмента? Если допустимы несколько вариантов вывода приведите их все. Варианты разделить словом «либо».
	char buf[2] = "12";
	int fd = creat("/a.txt", 0777);
	write(fd, buf, 2);
		12 или 21

№	Условие	Ответ

	close(fd);
	fd = open("/a.txt", O_RDONLY);
	fork();
	read(fd, buf, 1);
	printf("%c", buf[0]);
	exit(0);	

4	Что выведет программа? Считаем, что все системные вызовы отрабатывают успешно. Подключение заголовочных файлов опущено.
	
	int main() {
	  int fd[2];
	  char z = 'a';
	  pipe(fd);
	  write(fd[1], &z, sizeof(z));
	  if (!fork()) {
	  read(fd[0], &z, sizeof(z));
	  z = 'b';
	  write(fd[1], &z, sizeof(z));
	  write(1, (char[]){z, '1', '\n'}, 3);
	  return 0;
	  }
	  write(1, (char[]){z, '2', '\n'}, 3);
	  read(fd[0], &z, sizeof(z));
		a2 c3 b1
		b1 a2 c3
		a2 b1 c3

№	Условие	Ответ

	z = 'c';
	write(fd[1], &z, sizeof(z));
	write(1, (char[]){z, '3', '\n'}, 3);
	wait(0);
	}	

5	Что выведет программа? Считаем, что все системные вызовы отрабатывают успешно. Подключение заголовочных файлов опущено.
	
	int main() {
	int fd[2];
	char z = 'a';
	pipe(fd);
	write(fd[1], &z, sizeof(z));
	if (!fork()) {
	read(fd[0], &z, sizeof(z));
	z = 'b';
	write(1, (char[]){z, '1', '\n'}, 3);
	write(fd[1], &z, sizeof(z));
	return 0;
	}
	write(1, (char[]){z, '2', '\n'}, 3);
	read(fd[0], &z, sizeof(z));
	z = 'c';
		a2 c3 b1
		b1 a2 c3
		a2 b1 c3

№	Условие	Ответ

	write(1, (char[]){z, '3', '\n'}, 3);
	write(fd[1], &z, sizeof(z));
	wait(0);
	}	

Maria Kazachuk, Igor Mashechkin, Ivan Popov

Test tasks for the course “Operating systems”: manual. – Moscow: MAKS Press, 2022. – 164 p.

ISBN 978-5-317-06863-9

The manual was prepared by the authors to support the course “Operating Systems”, read in the third semester at the faculty of the CMC of Moscow State University, and is intended to test students' knowledge and prepare for the exam. This manual offers a combination of theoretical questions and tasks for programming in the C language according to the program of the lecture course.

The manual was developed on the basis of a basic set of test items and their modifications prepared by lecturers of the CMC MSU faculty: Volkova I.A., Vylitok A.A., Glazkova V.V., Gomzin A.G., Zhukov K.A., Kazachuk M.A., Kornykhin E.V., Kuzina L.N., Sanzharov V.V., Tyulyaeva V.V., Chernov A.V.

Keywords: operating systems, processes, process interaction, process execution planning.

Учебно-методическое издание

КАЗАЧУК Мария Андреевна
МАШЕЧКИН Игорь Валерьевич
ПОПОВ Иван Сергеевич

ТЕСТОВЫЕ ЗАДАНИЯ ПО КУРСУ «ОПЕРАЦИОННЫЕ СИСТЕМЫ»

Учебно-методическое пособие

Издательство «МАКС Пресс»
Главный редактор*: Е.М. Бугачева*

Напечатано с готового оригинал-макета

Подписано в печать 14.10.2022 г.
Формат 60х 90 1/16. Усл.печ.л. 10,25. Тираж 100 экз. Заказ 134.

Издательство ООО “МАКС Пресс”.
Лицензия ИД N 00510 от 01.12.99 г.

119992, ГСП-2, Москва, Ленинские горы, МГУ имени М.В. Ломоносова,
2-й учебный корпус, 527 к.
Тел. 8(495) 939-3890/91, Тел./Факс 8(495) 939-3891.

Отпечатано в полном соответствии с качеством
предоставленных материалов в ООО «Фотоэксперт»
115201, г. Москва, ул. Котляковская, д.3, стр. 13.