МИНОБРНАУКИ РОССИИ
Юго-Западный государственный университет

УТВЕРЖДАЮ:
Заведующий кафедрой
вычислительной техники

_____ И.Е. Чернецкая
«31» 08 2022 г.

ОЦЕНОЧНЫЕ СРЕДСТВА
для текущего контроля успеваемости
и промежуточной аттестации обучающихся
по дисциплине

Технология программирования

(наименование дисциплины)

09.03.02 Информационные технологии в бизнесе
(код и наименование ОПОП ВО)

Курск -2022

Задания для проведения текущего контроля успеваемости

Юго-Западный государственный университет

Вопросы для собеседования

Раздел 1. Понятие технологии программирования и основные этапы её развития

1. Что такое технологичность?
2. Какие технологии программирования вам известны?
3. В чем состоит сущность структурного программирования?
4. Что понимается под модульным программированием?
5. Назовите основные особенности объектной технологии программирования.
6. Что понимается под компонентной технологией программирования?
7. Что такое CASE-технологии?

Раздел 2. Проблемы разработки сложных программных систем. Блочно -иерархический подход к проектированию ПО

1. Какие факторы увеличивают сложность современных программных систем?
2. Почему при проектировании сложных программных систем стремятся пол у-чить иерархическую структуру?
3. Что понимается под повторным использованием кода?
4. К чему приводит наличие горизонтальных связей в структуре программной системы?
5. Что такое декомпозиция программной системы?
6. В чем состоит отличие декомпозиции программных систем по функциям и по данным?
7. Какова сущность объектной декомпозиции программных систем?

Раздел 3. Жизненный цикл и этапы разработки программного обеспечения. Эволюция м о-делей жизненного цикла.

1. Понятие жизненного цикла программы.
2. Фазы жизненного цикла.
3. Какой из этапов жизненного цикла ПО наиболее трудоемок при использовании классической технологии программирования?
4. Входит ли сопровождение в жизненный цикл ПО?
5. Верно ли, что жизненный цикл начинается с момента оформления техническ о-го задания на разработку ПО?
6. При использовании каких технологий программирования отладка программ не является самым трудоемким этапом разработки?
7. Верно ли, что жизненный цикл завершается в момент ввода ПО в эксплуат а-цию?

Раздел 4. Ускорение разработки ПО. Технология RAD

1. Какова основная цель применения технологии RAD?
2. Что принято понимать под прототипом программной системы?
3. Назовите этапы проектирования программной системы при использовании технологии RAD.
4. Что такое «функциональная точка» приложения в RAD?

5. Каким образом определяется необходимая численность коллектива разработчиков программной системы при использовании технологии RAD?
6. На какие классы программных систем ориентирована технология RAD?
7. Для каких программных систем использование технологии RAD является недопустимым?

Раздел 5. Определение требований к программному обеспечению. Техническое задание на разработку ПО.

1. Как определить требования к ПО
2. Каковы условия завершения подготовки технического задания?
3. Назовите базовые конструкции ТЗ.
4. Сформулируйте основные требования к ПО.
5. Какова связь требованиями к ПО и ТЗ?
6. Какова связь ТЗ с разработчиком?
7. Кто выступает заказчиком ПО и должен ли он владеть приемами программирования?

Раздел 6. Приемы повышения технологичности ПО. Модули и их свойства

Раздел 7. Нисходящее проектирование ПО. Структурное программирование.

1. Как влияет наличие горизонтальных межмодульных связей на качество нисходящего проектирования?
2. Каковы условия завершения итерационного процесса нисходящего проектирования программной системы?
3. Назовите базовые конструкции структурного программирования.
4. Сформулируйте теорему о структурировании программ.
5. Какова связь структурированности программ с их эффективностью?
6. Какова связь структурированности программ с их надежностью?
7. Почему «цикл с прерыванием» не является конструкцией структурного программирования?

Раздел 8. Проектирование программных систем. Нотации проектирования ПО. Функциональные диаграммы. Диаграммы потоков данных.

Раздел 9. Реализация программ. Стиль программирования. Оптимизация программ по времени выполнения и затратам памяти.

1. В чем отличие нисходящего подхода к проектированию программных систем от восходящего?
2. Какие нотации описания структурированных программ вам известны?
3. Назовите основные преимущества и недостатки языка граф-схем алгоритмов как одной из нотаций проектирования ПО.
4. Можно ли описать неструктурированный алгоритм на псевдокоде? Почему?
5. В чем заключаются преимущества псевдокода перед граф-схемой алгоритма?
6. Опишите назначение связей блока функциональной диаграммы.
7. Какие типы влияния блоков друг на друга предусмотрены в функциональных диаграммах?

Раздел 10. Методы отладки и тестирования программ. Интеграция программных систем.

1. Какие метода отладки существуют?
2. Методы тестирования.
3. Какими частными характеристиками определяется тестирование программных систем?
4. Каковы основные источники тестирования ПО?
5. Что понимается под интеграцией программы?

6. Как реализуют интеграцию программной системы?
7. Когда программная система считается тестируемой?

Раздел 11. Организация коллективной разработки программных систем. Управление разработкой, планирование, оценка трудоемкости.

1. Каковы преимущества коллективной разработки программных систем?
2. Почему коллективная разработка ПО остается эффективной только до определенного максимального числа исполнителей?
3. Что понимается под бригадной разработкой ПО?
4. Какие варианты организации бригад разработчиков вам известны?
5. Как распределяются роли участников в бригаде главного программиста?
6. В каких случаях целесообразно переходить к демократической бригаде?
7. В каких единицах измеряется трудоемкость разработки ПО?

Раздел 12. Модели надежности ПО, методы оценки и повышения надежности программных средств.

1. Какими частными характеристиками определяется надежность программных систем?
2. Каковы основные источники снижения надежности ПО?
3. Что понимается под стабильностью программы?
4. Как определяется устойчивость программной системы?
5. Когда программная система считается восстанавливаемой?
6. Что такое отказ?
7. Чем отказ отличается от сбоя?

Критерии оценки:

- 0 баллов выставляется обучающемуся, если студент не может ответить на поставленные вопросы или допустил принципиальные ошибки в выполнении предусмотренных программой знаний.
- 1 балла выставляется обучающемуся, если доля правильных ответов от 50% до 80%.
- 2 балла выставляется обучающемуся, если доля правильных ответов более 80%.

Составитель  В.С. Панищев

«29» августа 2019 г

Юго-Западный государственный университет

Кафедра информационных систем и технологий

Вопросы к защите лабораторных работ
по дисциплине «Технология разработки программного обеспечения»

1. Каковы особенности RAD-подхода при разработке прикладного ПО?
2. Каким образом определяются метод и технология проектирования ПО?
3. Какие стандарты необходимы для выполнения конкретного проекта?
4. В чем заключаются основные принципы структурного подхода?
5. Что общего и в чем различия между методом SADT и моделированием потоков данных?
6. В чем заключаются достоинства и недостатки структурного подхода?
7. Проектирование архитектуры ПС.
8. Какие модели предпроектного исследования используются на этапе проектирования и для чего?
9. Что является результатом этапа проектирования ПС?
10. Объектно-ориентированный подход к проектированию программного обеспечения.
11. В чем заключаются основные принципы объектно-ориентированного подхода?
12. Что общего и в чем различия между структурно-функциональным и объектно-ориентированным подходом?
13. В чем заключаются достоинства и недостатки объектно-ориентированного подхода?
14. В чем заключаются достоинства и недостатки структурно-функционального подхода?
15. Как влияет наличие горизонтальных межмодульных связей на качество нисходящего проектирования?
16. Каковы условия завершения итерационного процесса нисходящего проектирования программной системы?
17. Назовите базовые конструкции структурного программирования.
18. Сформулируйте теорему о структурировании программ.
19. Почему «цикл с прерыванием» не является конструкцией структурного программирования?
20. Какие вы знаете нотации нисходящего проектирования, которые не позволяют описывать неструктурированные алгоритмы?
21. Какая программа является более технологичной – структурированная или не структурированная? Почему?
22. Что такое тестирование программы?
23. Чем отличается процесс тестирования от процесса отладки?
24. Перечислите принципы тестирования.
25. Какие методы тестирования вы знаете?
26. Какие виды ошибок вы знаете?
27. Когда должна заканчиваться стадии тестирования и отладки ПО?

Критерии оценки:

Форма контроля	Минимальный балл		Максимальный балл	
	балл	примечание	балл	примечание

Форма контроля	Минимальный балл		Максимальный балл	
	балл	примечание	балл	примечание
Защита лабораторной «Формулирование требований к проектируемому ПО. Разработка технического задания»	2	Выполнение, доля правильных ответов от 50% до 80%	4	Выполнение, доля правильных ответов более 80%
Защита лабораторной «Модули и их свойства. Связность и межмодульное сцепление.»	2	Выполнение, доля правильных ответов от 50% до 80%	4	Выполнение, доля правильных ответов более 80%
Защита лабораторной «Нисходящее проектирование ПО. Структурное программирование»	2	Выполнение, доля правильных ответов от 50% до 80%	4	Выполнение, доля правильных ответов более 80%
Защита лабораторной «Проектирование программных систем. Функциональные диаграммы. Диаграммы потоков данных.»	3	Выполнение, доля правильных ответов от 50% до 80%	6	Выполнение, доля правильных ответов более 80%
Защита лабораторной «Реализация программных систем. Отладка и тестирование. Стиль программирования»	3	Выполнение, доля правильных ответов от 50% до 80%	6	Выполнение, доля правильных ответов более 80%

Составитель В.С. Панищев

«29» августа 2019 г

Юго-Западный государственный университет

Кафедра информационных систем и технологий

Типовые задания для проведения промежуточной аттестации

В закрытой форме:

1. Межмодульное сцепление представляет собой:

Вариант 1: меру взаимозависимости модулей, которая определяет, насколько хорошо модули изолированы друг от друга
Вариант 2: меру взаимозависимости модулей, которая определяет, насколько хорошо модули связаны друг с другом
Вариант 3: меру взаимозависимости модулей, которая определяет степень повторного использования кода одного модуля в другом моду-лей
Вариант 4: меру взаимозависимости модулей, которая определяет число разделяемых глобальных переменных, используемых модулями
Вариант 5: меру взаимозависимости модулей, которая определяет число разделяемых файлов, используемых модулями

2. Какое из перечисленных ниже требований и условий наиболее существенно для выпол-нения отладки программы?

Варианты ответа:
Вариант 1: до перехода к отладке необходимо во всех деталях понять зада-чу, решаемую программой
Вариант 2: до перехода к отладке необходимо исправить все ошибки разра-ботанной программной документации
Вариант 3: до перехода к отладке необходимо устранить все технологиче-ские ошибки в тексте программы
Вариант 4: отладка программы может выполняться без непосредственного ее выполнения на компьютере
Вариант 5: отладка программы может выполняться без использования средств автоматизации

3. Какие из перечисленных ниже факторов определяют временную эффективность про-граммы: временная сложность алгоритма решения задачи; количество файлов в проекте программы; правила выбора идентификаторов для программных сущностей; количество и число итераций циклов программы; число глобальных переменных в про-грамме.

Вариант 1: первый и четвертый
Вариант 2: первый и пятый
Вариант 3: только четвертый
Вариант 4: только первый
Вариант 5: только пятый

4. Какое из сформулированных ниже утверждений верно?

Вариант 1: сложность поиска и устранения алгоритмических ошибок в программе максимальна среди всех видов ошибок
Вариант 2: исправление алгоритмических ошибок не требует возврата к этапу проектирования программной системы
Вариант 3: технологические ошибки вносятся на этапах определения тре-бований к компонентам программной системы и их проекти-рования

Вариант 4: исправление ошибок программирования возможно только с использованием методов белого ящика
Вариант 5: исправление ошибок программирования возможно только с использованием методов черного ящика

5. Какое из сформулированных ниже утверждений неверно?

Вариант 1: наработка программы до отказа не зависит от числа не выяв-ленных ошибок проектирования
Вариант 2: наработка программы до отказа не зависит от длины иденти-фикаторов, используемых в программе
Вариант 3: наработка программы до отказа не зависит от числа исправлен-ных технологических ошибок
Вариант 4: наработка программы до отказа может быть увеличена при вве-дении временной избыточности
Вариант 5: наработка программы до отказа может быть увеличена при вве-дении информационной избыточности

6. Отладка программы представляет собой:

Вариант 1: процесс выполнения программы на контрольных данных с це-лью поиска ошибок и устранения их причин
Вариант 2: процесс выполнения программы на контрольных данных с це-лью поиска ошибок и устранения их симптомов
Вариант 3: процесс выполнения программы на контрольных данных с це-лью доказательства отсутствия ошибок
Вариант 4: процесс выполнения программы на контрольных данных с це-лью накопления статистических данных об ошибках
Вариант 5: процесс выполнения программы на контрольных данных с це-лью разработки тестов

7. Проектирование программной системы имеет целью:

Вариант 1: разработку внутренней структуры программной системы без детализации компонент
Вариант 2: разработку внутренней структуры программной системы и ин-терфейса пользователя
Вариант 3: разработку внутренней структуры программной системы и ос-новных алгоритмов
Вариант 4: разработку внутренней структуры программной системы с де-тализацией компонент
Вариант 5: разработку внутренней структуры программной системы и ос-новным форматов данных

8. Прочность модуля определяет:

Вариант 1: насколько сильно программные и информационные элементы модуля связаны между собой
Вариант 2: насколько сильно программные и информационные элементы модуля связаны с элементами других модулей
Вариант 3: насколько сильно программные и информационные элементы модуля связаны с главным модулем программной системы
Вариант 4: насколько сильно программные и информационные элементы модуля связаны со стандартными модулями
Вариант 5: насколько сильно программные и информационные элементы модуля связаны со стандартными библиотеками

9. Какое из сформулированных ниже утверждений неверно?

Вариант 1: информационно прочный модуль наименее технологичен
Вариант 2: функционально прочный модуль является наиболее техноло-гичным
Вариант 3: для библиотек ресурсов характерна логическая прочность
Вариант 4: чем выше прочность модулей, тем меньше межмодульное сцеп-ление
Вариант 5: чем выше сцепление модулей, тем меньше их прочность

10. Какое из сформулированных ниже утверждений верно?

Вариант 1: тестирование является одной из фаз отладки программы
Вариант 2: тестирование является доказательством правильности програм-мы
Вариант 3: тестирование не требует наличия эталона
Вариант 4: зависимость времени полного тестирования программы от объ-ема входных данных линейная
Вариант 5: зависимость времени полного тестирования программы от объ-ема входных данных линейно-логарифмическая

11. Какое из сформулированных ниже утверждений неверно?

Вариант 1: коэффициент готовности определяет вероятность правильного функционирования программы
Вариант 2: коэффициент готовности программы зависит от числа не выяв-ленных ошибок проектирования
Вариант 3: коэффициент готовности программы повышается при введении в программу структурной избыточности
Вариант 4: коэффициент готовности программы определяет вероятность того, что программа в данный момент работоспособна
Вариант 5: коэффициент готовности программы является количественной характеристикой надежности программы

12. Какое из сформулированных ниже утверждений верно?

Вариант 1: повторное использование кода способствует повышению надежности программы
Вариант 2: наработка программы на отказ измеряется в числе операторов
Вариант 3: коэффициент готовности программы измеряется в единицах времени
Вариант 4: восстанавливаемость программы является количественной ха-рактеристикой
Вариант 5: повторное использование кода возможно только при объектном программировании

13. Какие из приведенных ниже идентификаторов отвечают критериям качества стиля про-граммирования?

__GetLastToken
abc123
__STDIO_H__
ab_plus

Вариант 1: первый и третий
Вариант 2: первый и второй
Вариант 3: второй и четвертый
Вариант 4: только первый
Вариант 5: только четвертый

14. Каким образом качество программной системы зависит от используемой технологии разработки?

Вариант 1: скачкообразно растет при переходе к более совершенной техно-логии разработки
Вариант 2: зависимости нет
Вариант 3: линейно возрастает при переходе к более совершенной техно-логии разработки
Вариант 4: полиномиально возрастает при переходе к более совершенной технологии разработки
Вариант 5: зависимость носит случайный характер

15. Верно ли, что более технологичная программа всегда является более эффективной по времени выполнения?

Вариант 1: нет, как правило, повышение технологичности программы снижает ее временную эффективность
Вариант 2: да, более технологичная программа всегда является более эф-фективной по времени выполнения
Вариант 3: все зависит от квалификации разработчика, который занимает-ся повышением технологичности
Вариант 4: все зависит от используемой технологии разработки программы
Вариант 5: все зависит от используемых сред и иных инструментов разра-ботки ПО

16. Какой тип межмодульного сцепления реализован в следующей функции? Function MinMax (a, b: integer; flag: boolean): integer;

```
begin
if(a>b) and (flag) then MinMax: =a
else MinMax: =b;
end;
```

Вариант 1: сцепление по данным и по управлению
Вариант 2: сцепление по данным
Вариант 3: сцепление по управлению
Вариант 4: сцепление по образцу
Вариант 5: сцепление по образцу и по управлению

17. Какой тип межмодульного сцепления реализован в следующей функции? Function Max-El(a:array of integer): integer;

```
Var i: word;
begin
MaxEl: =a [0];
for i: =l to High (a) do
if a [i]>MaxEl then MaxEl: =a [i];
end;
```

Вариант 1: сцепление по образцу
Вариант 2: сцепление по данным
Вариант 3: сцепление по общей области данных
Вариант 4: сцепление по управлению
Вариант 5: сцепление по образцу и по данным

18. Какие функции называют информационно связанными?

Вариант 1: функции, которые обрабатывают одни и те же данные
Вариант 2: функции, которые обрабатывают данные одного и того же формата
Вариант 3: функции, которые обрабатывают данные одного и того же типа
Вариант 4: функции, которые входят в одну и ту же библиотеку
Вариант 5: функции, которые принадлежат одному и тому же модулю

19. Какой тип связности характерен для функций, входящих в библиотеки?
Вариант 1: логическая связность
Вариант 2: информационная связность
Вариант 3: случайная связность
Вариант 4: процедурная связность
Вариант 5: функциональная связность

20. Что сильнее всего ухудшает эффективность программы по времени выполнения?
Вариант 1: наличие многократно вложенных циклов
Вариант 2: наличие большого числа многоальтернативных ветвлений
Вариант 3: наличие большого числа статических массивов
Вариант 4: наличие обращений к элементам многомерных массивов
Вариант 5: наличие рекурсивных функций

21. Какое из сформулированных ниже утверждений неверно?
Вариант 1: создание программной системы начинается с разработки ин-терфейса пользователя
Вариант 2: создание программной системы завершается оформлением ком-плекта документации
Вариант 3: создание программной системы начинается с предпроектных исследований и сбора информации
Вариант 4: создание программной системы включает последовательность этапов, порядок следования которых зависит от выбранной мо-дели жизненного цикла
Вариант 5: создание программной системы может осуществляться коллек-тивом разработчиков при условии ее правильной декомпози-ции

22. Какие из перечисленных ниже факторов определяют надежность программы?
Вариант 1: число не выявленных ошибок проектирования и используемая технология разработки
Вариант 2: только используемая технология разработки
Вариант 3: только число не выявленных ошибок проектирования
Вариант 4: длина идентификаторов
Вариант 5: число используемых препроцессорных директив

23. Что понимают под технологичностью программного средства?
Вариант 1: качество проекта программного средства, от которого зависят трудовые и материальные затраты на его реализацию и после-дующие модификации
Вариант 2: качество проекта программного средства, обеспечивающее упрощение его отладки и тестирования
Вариант 3: качество проекта программного средства, от которого зависят финансовые затраты на его реализацию и последующие моди-фикации
Вариант 4: степень соответствия проекта программного средства государ-ственным и отраслевым стандартам
Вариант 5: степень соответствия проекта программного средства корпора-тивным стандартам

24. Что принято называть универсальностью программы?
Вариант 1: обеспечение правильной работы программы при любых допу-стимых данных и защиты от неправильных данных
Вариант 2: обеспечение устойчивой работы программы при любых допу-стимых данных и защиты от неправильных данных

Вариант 3: обеспечение эффективной работы программы при любых допу-стимых данных и защиты от неправильных данных
Вариант 4: обеспечение правильной работы программы при любых данных и защиты от неправильных данных
Вариант 5: обеспечение правильной работы программы при любых допу-стимых данных и защиты от неправильных входных данных

25. Какие программы называют защищенными?
Вариант 1: которые обеспечивают конфиденциальность информации
Вариант 2: которые функционируют устойчиво при внешних аномалиях
Вариант 3: которые устойчивы к воздействию вирусов
Вариант 4: которые устойчивы к взаимодействию с другими программами
Вариант 5: которые защищены от копирования

26. Что такой адаптируемость программы?
Вариант 1: возможность быстрой модификации с целью приспособления к изменяющимся условиям функционирования
Вариант 2: возможность быстрой модификации с целью приспособления к новой аппаратной конфигурации системы
Вариант 3: возможность быстрой модификации при обнаружении ошибок
Вариант 4: возможность быстрой модификации по требованию пользова-теля
Вариант 5: возможность быстрой настройки под конкретную задачу

27. Какой из этапов проектирования программной системы характеризуется наибольшей трудоемкостью?
Вариант 1: отладка, компоновка и тестирование программной системы
Вариант 2: анализ требований к программной системе
Вариант 3: проектирование программной системы и ее компонентов
Вариант 4: алгоритмизация и реализация компонентов
Вариант 5: разработка проектной документации

28. Какая из следующих операций является операцией сравнения двух переменных?
Вариант 1: ==
Вариант 2: :=
Вариант 3: equal
Вариант 4: =
Вариант 5: нет правильного варианта

29. Какой из ниже перечисленных операторов, НЕ является циклом в С++?
Вариант 1: repeat until
Вариант 2: do while
Вариант 3: while
Вариант 4: for
Вариант 5: нет правильного варианта

30. Какую функцию должны содержать все программы на С++?
Вариант 1: main()
Вариант 2: program()
Вариант 3: start()
Вариант 4: system()
Вариант 5: нет правильного ответа

31. Какой из перечисленных типов данных не является типом данных в C++?
Вариант 1: real
Вариант 2: int
Вариант 3: double
Вариант 4: float
Вариант 5: нет правильного ответа

32. Каков будет результат выражения «!(1 && !(0 || 1))» ?
Вариант 1: True
Вариант 2: Неоднозначность
Вариант 3: False
Вариант 4: Среди предложенных вариантов нет верного

33. Что будет выведено на экран в результате выполнения этого фрагмента кода?
Вариант 1: 1 = 2
Вариант 2: a = b
Вариант 3: Синтаксическая ошибка
Вариант 4: Вывод на экран не выполнится

34. Какой из следующих логических операторов - логический оператор «И»?
Вариант 1: &&
Вариант 2: &
Вариант 3: |
Вариант 4: ||

35. Это значение 5.9875e17 может быть сохранено в переменной, типа:
Вариант 1: float
Вариант 2: int
Вариант 3: long
Вариант 4: long long
Вариант 5: нет правильного ответа

36. У какой из представленных операций самый высокий приоритет выполнения?
Вариант 1: !
Вариант 2: &&
Вариант 3: ||
Вариант 4: Операции имеют одинаковый приоритет

37. Какой заголовочный файл следует подключить, чтобы можно было пользоваться приведением типов данных?
Вариант 1: Никакого
Вариант 2: cmath
Вариант 3: cctype
Вариант 4: alg

38. Результат выполнения следующего фрагмента кода«!((1 || 0) && 0)»?
Вариант 1: 1
Вариант 2: 0
Вариант 3: Результат не может быть заранее определен
Вариант 4: Среди ответов нет верного

39. Укажите операцию, приоритет выполнения которой выше чем у остальных:

Вариант 1: ++
Вариант 2: /
Вариант 3: +
Вариант 4: *

40. Результат выполнения следующего фрагмента кода: «54 << 3»?
Вариант 1: 432
Вариант 2: 57
Вариант 3: 54000
Вариант 4: 543

41. Какое ключевое слово указывает, что целая переменная НЕ может принимать отрицательные значения?
Вариант 1: unsigned
Вариант 2: nonnegative
Вариант 3: positive
Вариант 4: нет правильного варианта

42. Что будет напечатано, после выполнения этого кода: «cout << (5 << 3);»?
Вариант 1: 40
Вариант 2: 53
Вариант 3: 35
Вариант 4: 5000
Вариант 5: нет правильного варианта

43. Какой из ниже перечисленных вариантов ответа, показывает правильно записанный оператор выбора if?
Вариант 1: if (условное выражение)
Вариант 2: условное выражение if
Вариант 3: if условное выражение
Вариант 4: if {условное выражение}
Вариант 5: нет правильного варианта

44. Какой порядковый номер последнего элемента массива, размер массива 19?
Вариант 1: 18
Вариант 2: 19
Вариант 3: 20
Вариант 4: Порядковый номер определяется программистом
Вариант 5: нет правильного варианта

45. В каком из вариантов ответов объявлен двумерный массив?
Вариант 1: int anarray[20][20]
Вариант 2: array anarray[20][20]
Вариант 3: int array[20, 20]
Вариант 4: char array[20]
Вариант 5: нет правильного варианта

46. Как правильно освободить память, после выполнения этого кода? { char *a; a = new char[20];}
Вариант 1: delete a[]
Вариант 2: delete [] a
Вариант 3: delete a

Вариант 4: Нет необходимости высвобождать память, это произойдет автоматически после выхода из операторного блока

47. Укажите правильное объявление указателя в C++
Вариант 1: int *x
Вариант 2: ptr x
Вариант 3: int &x
Вариант 4: int x

48. Укажите зарезервированное ключевое слово для динамического выделения памяти
Вариант 1: malloc
Вариант 2: realloc
Вариант 3: dynamic_cast
Вариант 4: dynamic
Вариант 5: нет правильного варианта

49. Какой заголовочный файл C++ содержит инструкции файлового ввода/вывода?
Вариант 1: fstream
Вариант 2: infstream
Вариант 3: iostream
Вариант 4: filestream
Вариант 5: нет правильного варианта

50. При определении структуры необходимо использовать следующее ключевое слово:
Вариант 1: struct
Вариант 2: enum
Вариант 3: structure
Вариант 4: record
Вариант 5: нет правильного ответа

51. В каком из следующих вариантов ответов выполнен корректный доступ к полю структуры, причём структура объявлена через указатель?
Вариант 1: b->var
Вариант 2: b>var
Вариант 3: b-var
Вариант 4: b.var
Вариант 5: нет правильного ответа

52. Какая из переменных хранит количество аргументов, передаваемых в программу?
Вариант 1: argc
Вариант 2: count
Вариант 3: arglen
Вариант 4: argv

53. Для чего используются встроенные функции?
Вариант 1: Для увеличения скорости работы программы
Вариант 2: Чтобы уменьшить размер программы
Вариант 3: Для распределения ресурсов системы
Вариант 4: Для удаления ненужных функций

54. Что из нижеперечисленного не является прототипом функции?
Вариант 1: funct((char)x);

Вариант 2: int funct(char x, char y);
Вариант 3: charx();
Вариант 4: void funct();

55. Выберите правильное (полное) определение функции.
Вариант 1: int funct(int x){return x = x + 1;}
Вариант 2: int funct();
Вариант 3: void funct(int){cout << "Hello"}
Вариант 4: void funct(x){cout << "Hello"}
Вариант 5: нет правильного ответа

56. Можно ли перегрузить функцию main()?
Вариант 1: Нельзя
Вариант 2: Можно
Вариант 3: Можно, соблюдая осторожность
Вариант 4: Можно, но лучше не надо

57. Что значит ключевое слово inline?
Вариант 1: Вместо вызова данной функции в точку вызова подставляется ее код (на усмотрение компилятора)
Вариант 2: Все вызовы встроенных функции заменяются вызовом этой функции (на усмотрение компилятора)
Вариант 3: Сообщает компилятору использовать функцию только в пределах одной единицы трансляции
Вариант 4: Среди предложенных вариантов нет верного

58. для использования std::shared_ptr необходимо подключить заголовочный файл :
Вариант 1: #include <memory>
Вариант 2: #include <iostream>
Вариант 3: #include <string>
Вариант 4: #include <stdio>

59. std::shared_ptr появился в стандарте C++:
Вариант 1: 2011 года
Вариант 2: 2014 года
Вариант 3: 2003 года
Вариант 4: 2017 года

60. std::vector <int > v1 (10); std::vector<int> v2 (15); поменять местами два объекта (v1 , v2) типа std::vector можно следующим образом :
Вариант 1: v1.swap (v2);
Вариант 2: v2 = v1;
Вариант 3: v1 != v2;
Вариант 4: v1 = v2;

61. Какие преимущества std::vector над динамически выделяемым массивом ?
Вариант 1: std::vector сам управляет памятью, увеличивая размер массива и удаляя его
Вариант 2: std::vector сам определяет тип данных, добавляемые в него и инициализирует их
Вариант 3: std::vector способен хранить переменные разных типов
Вариант 4: преимуществ нет, выполняет те же действия, что и динамически выделяемый массив

62. void foo (const Train& head); аргумент head в функции foo является:
Вариант 1: константной ссылкой
Вариант 2: константным указателем
Вариант 3: константной копией объекта head
Вариант 4: лямбда-выражением для вызова функции foo

63. Часть общего объявления функции, позволяющая средствам трансляции идентифицировать функцию среди других, называется:
Вариант 1: Сигнатурой функции
Вариант 2: Телом функции
Вариант 3: Функциональным блоком
Вариант 4: Ни один из предложенных вариантов

64. Описание того, что данная функция делает, это:
Вариант 1: Семантика функции
Вариант 2: Сигнатура функции
Вариант 3: Функциональный блок
Вариант 4: Ни один из предложенных вариантов

65. Какое из следующих утверждений верно?
Вариант 1: Компилятор может проигнорировать объявление встроенной функции
Вариант 2: Встроенные функции не могут возвращать значения
Вариант 3: Встроенные функции должны возвращать значение
Вариант 4: Встроенные функции не должны содержать более определенного количества строк кода (определяется компилятором)

66. Какие из следующих функций являются встроенными?
Вариант 1: inline void foo() {}
Вариант 2: void foo() inline {}
Вариант 3: Нет правильного ответа
Вариант 4: inline:void foo() {}

67. Сколько аргументов может иметь деструктор?
Вариант 1: Не имеет аргументов
Вариант 2: Любое количество
Вариант 3: Не менее одного
Вариант 4: Один

68. Какого спецификатора доступа в классах нет?
Вариант 1: remote
Вариант 2: protected
Вариант 3: public
Вариант 4: private

69. Укажите корректное объявление класса:
Вариант 1: class A { int x; };
Вариант 2: object A { int x; };
Вариант 3: public class A { }
Вариант 4: class B { }
Вариант 5: нет правильного варианта

70. Какое значение должен возвращать деструктор?
Вариант 1: деструкторы не возвращают значение
Вариант 2: объект класса
Вариант 3: код состояния о правильном удалении класса
Вариант 4: указатель на класс
Вариант 5: нет правильного варианта

71. Что такое деструктор?
Вариант 1: Деструктор - это специальная функция-элемент, которая должна уничтожать экземпляр класса после завершения его работы
Вариант 2: Деструктор - это специальная функция-элемент, которая должна отслеживать данные в экземпляре класса в процессе работы
Вариант 3: Деструктор - это функция, которая должна открывать динамическую область для экземпляра класса
Вариант 4: Деструктор - это функция, которая обнуляет значения передаваемых в нее параметров
Вариант 5: нет правильного варианта

72. Что выведется на экран в результате выполнения данного кода? {int a = 8; int b = a++; int c = ++b; int res = ++c+b---a; std::cout << res;}
Вариант 1: 10
Вариант 2: 9
Вариант 3: 11
Вариант 4: 12
Вариант 5: нет правильного варианта

73. Что выведется на экран в результате выполнения данного кода? { const int a = 8; a = 3; std::cout << a; }
Вариант 1: Ничего, возникнет ошибка компиляции
Вариант 2: 3
Вариант 3: 8
Вариант 4: Среди ответов нет верного

74. Что выведется на экран в результате выполнения данного кода? {const int a = 8; int b = 7; int c = a>b?a:b; std::cout << c;}
Вариант 1: 8
Вариант 2: 7
Вариант 3: Ничего, возникнет ошибка компиляции
Вариант 4: Среди ответов нет верного

75. Оператор switch позволяет:
Вариант 1: Осуществить выбор среди нескольких фрагментов кода
Вариант 2: Установить атрибуты текущей сессии
Вариант 3: Переключиться между режимами чтения-записи данных
Вариант 4: Подключить дополнительные библиотеки

76. Оператор goto:
Вариант 1: Безусловно передаёт управление в инструкцию с меткой заданным идентификатором
Вариант 2: Передает управление в инструкцию с меткой заданным идентификатором с учетом выполнения проверки

Вариант 3: В языке C++ не делает ничего, так как морально устарел и не был включен в стандарт
Вариант 4: Передает управление в функцию main
Вариант 5: нет правильного варианта

77. Основные типы в C++ делятся на несколько категорий. Из предложенного списка выберите лишнюю:
Вариант 1: Лишних нет
Вариант 2: void
Вариант 3: С плавающей запятой
Вариант 4: Целочисленные

78. Указатель this доступен только в нестатических функциях-членах типа class, struct или union. Он указывает на:
Вариант 1: Объект, для которого вызывается функция-член
Вариант 2: Объект родительского класса для класса, в котором вызывается функция-член
Вариант 3 Конструктор объекта
Вариант 4: Первое неконстантное поле класса (первое указанное при объявлении класса)
Вариант 5: нет правильного варианта

79. Объединения (union) могут быть полезны для:
Вариант 1: Экономии памяти
Вариант 2: Объединения участков памяти из стека и кучи (heap, stack)
Вариант 3: Конкатенации значений переменных разных типов
Вариант 4: Для приведения значений переменных к строковому типу
Вариант 5: нет правильного варианта

80. Если указатель имеет тип void *, то:
Вариант 1: Он может указывать на любую переменную, объявленную без указания ключевого слова const или volatile
Вариант 2: Это указатель на свободный участок памяти, данные в котором сохранены и могут быть перезаписаны
Вариант 3: Он указывает на адрес памяти, по которому расположена функция main
Вариант 4: Он может указывать на переменную пользовательского типа, объявленную с модификатором const
Вариант 5: нет правильного варианта

81. Оператор разрешения области :: используется для:
Вариант 1: Для устранения неоднозначности идентификаторов
Вариант 2: Не содержится в стандарте языка C++
Вариант 3: Для выхода из глубоко вложенного цикла
Вариант 4: Выбора подходящей функции из списка перегруженных

82. Имя функции должно начинаться с:
Вариант 1: Символа подчеркивания или буквы
Вариант 2: Символа подчеркивания
Вариант 3: Символа подчеркивания, буквы или цифры
Вариант 4: Символа подчеркивания, буквы, цифры или символа «^»

83. Выберите правильный вариант записи абстрактного класса в C ++ :

Вариант 1: class A { public: virtual int f () = 0;};
Вариант 2: abstract class A { public: virtual int f () = 0;};
Вариант 3: class A { public: virtual int f () = 0;} abstract;
Вариант 4: class A { public: virtual int f ();};

84. Инкапсуляция — это:
Вариант 1: ограничение доступа к определённым полям и методам класса
Вариант 2: разбиение логически обособленных модулей на классы
Вариант 3: изолирование методов в отдельный класс, полей в структуры
Вариант 4: создание класса-оболочки над используемым классом

85. Выберите правильное обращение к методу foo() класса someClass:
Вариант 1: pointer→foo();
Вариант 2: pointer.foo();
Вариант 3: pointer::foo();
Вариант 4: someClass::foo();

86. ключевое слово volatile в C++ :
Вариант 1: информирует компилятор что переменная может быть изменена НЕявным способом т.е. без явного использовать оператора присвоения
Вариант 2: информирует компилятор что переменная НЕ может быть изменена НЕявным способом т.е. без явного использовать оператора присвоения
Вариант 3: информирует компилятор что переменная НЕ может быть изменена Явным способом т.е. с явным использованием оператора присвоения
Вариант 4: отсутствует

87. ключевое слово mutable в C++:
Вариант 1: позволяет совершать изменения над объектом в константном методе
Вариант 2: модифицирует поле класса в константное, запрещающее изменения поля
Вариант 3: отсутствует
Вариант 4: отменяет действия модификатора volatile

88. ключевое слово auto в C++ стандарта 2011 года :
Вариант 1: позволяет не указывать тип переменной явно, говоря компилятору, чтобы он сам определил фактический тип переменной, на основе типа инициализируемого значения или типа возвращаемого значения
Вариант 2: указывается перед объявлением объекта, говоря компилятору,чтобы она сам управлял временем жизни объекта
Вариант 3: автоматически инициализирует все поля объекта класса
Вариант 4: отсутствует

89. В системе контроля версий git команда checkout :
Вариант 1: переключает ветвь с текущей на указанную
Вариант 2: сохраняет изменения в журнале изменений
Вариант 3: создает ветвь
Вариант 4: отключается от текущей ветки и удаляет её

90. В системе контроля версий git команда branch:
Вариант 1: создает ветвь
Вариант 2: отключается от текущей ветки и удаляет её
Вариант 3: переключает ветвь с текущей на указанную
Вариант 4: сохраняет изменения в журнале изменений

91. std::string представляет собой :
Вариант 1: класс оболочка, хранящий внутри строку и предоставляющий методы для работы со строкой
Вариант 2: typedef const char* string; в пространстве имён std
Вариант 3: одномерный массив, хранящий строку данных
Вариант 4: зарезервированное слово в пространстве имён std

92. В C++ копирующий конструктор в классе:
Вариант 1: можно переопределить
Вариант 2: необходимо переопределить
Вариант 3: запрещено переопределять
Вариант 4: отсутствует

93. ключевое слово explicit в C++
Вариант 1: спецификатор метода, указывающий запрет на неявное приведение типов
Вариант 2: спецификатор метода, указывающий на уникальность и приоритет выполняемого метода
Вариант 3: спецификатор поля, позволяющий контролировать срок жизни поля класса в случае работы с массивами(одномерными, двумерными или многомерными)
Вариант 4: отсутствует

94. Callback-функция:
Вариант 1: функция, передаваемая в другую функцию\метод, и вызываемая при определенных условиях или по достижению требуемого участка кода
Вариант 2: функция, которая вызывает переданную функцию как аргумент
Вариант 3: метод или функция, вызываемая из динамической библиотеки
Вариант 4: метод или функция, вызываемая из статической библиотеки

95. Функтор – это :
Вариант 1: класс, в котором переопределен оператор ()
Вариант 2: класс, содержащий виртуальные методы, но не имеющий полей
Вариант 3: пространство имён, содержащее функции
Вариант 4: заголовочный файл, содержащий прототипы функций

96. Выберите правильный вариант написания лямбда-выражения:
Вариант 1: auto lambda = [](auto a, auto b) { return a < b; };
Вариант 2: auto lambda = [][auto a, auto b] { return a < b; };
Вариант 3: auto lambda = ()[auto a, auto b] { return a < b; };
Вариант 4: auto lambda = [](auto a, auto b) [{ return a < b; }];

97. При написании классов, нахождение полей в секции private, а методов доступа к ним в секции public называется :
Вариант 1: инкапсуляцией
Вариант 2: интерполяцией
Вариант 3: интерпретацией
Вариант 4: безопасностью доступа к полям класса на уровне методов класса

98. Использование структур внутри класса :
Вариант 1: ограничений нет
Вариант 2: ни один из предложенных вариантов
Вариант 3: запрещено, разрешено иметь только ссылки на структуры

Вариант 4: запрещено, разрешено иметь только указатели на структуры

99. Последовательность жизненного цикла программного обеспечения (ПО) представляет собой:

Вариант 1: период времени от момента зарождения идеи создания ПО до момента завершения его поддержки

Вариант 2: период времени от момента зарождения идеи создания ПО до момента его внедрения в эксплуатацию

Вариант 3: период времени от момента подписания технического задания на создание ПО до момента завершения его поддержки

Вариант 4: период времени от момента подписания технического задания на создание ПО до момента оформления эксплуатационной до-кументации

Вариант 5: период времени от момента подписания технического задания на создание ПО до момента завершения его эксплуатации

100. Какое из сформулированных ниже утверждений верно?

Вариант 1: декомпозиция программной системы на модули позволяет орга-низовать параллельную работу над несколькими модулями

Вариант 2: декомпозиция программной системы по функциональному принципу более эффективна, чем декомпозиция по потокам данных

Вариант 3: декомпозиция программной системы всегда приводит к иерар-хической структуре

Вариант 4: декомпозиция программной системы усложняет процесс проек-тирования, но повышает надежность программы

Вариант 5: декомпозиция программной системы усложняет повторное ис-пользование кода

101. Какое из сформулированных ниже утверждений неверно?

Вариант 1: сцепление модулей по образцу не хуже сцепления по данным с точки зрения модифицируемости

Вариант 2: сцепление модулей по образцу не хуже сцепления по данным с точки зрения наглядности

Вариант 3: сцепление модулей по образцу лучше сцепления по управле-нию с точки зрения возможности повторного использования кода

Вариант 4: сцепление модулей по образцу не уступает сцеплению по управлению с точки зрения устойчивости к ошибкам в других модулях

Вариант 5: сцепление модулей по образцу уступает сцеплению по управ-лению с точки зрения сложности документирования

102. Какое из сформулированных ниже утверждений верно?

Вариант 1: прямоугольником в схемах алгоритмов, программ и систем обозначается произвольный процесс обработки данных

Вариант 2: прямоугольником в схемах алгоритмов, программ и систем обозначается процесс вывода данных на бумажный носитель

Вариант 3: прямоугольником в схемах алгоритмов, программ и систем обозначается процесс принятия решений

Вариант 4: прямоугольником в схемах алгоритмов, программ и систем обозначаются циклические процессы

Вариант 5: прямоугольником в схемах алгоритмов, программ и систем обозначаются предопределенные процессы

103. Какое из сформулированных ниже утверждений неверно?

Вариант 1: тестирование программы методом покрытия операторов более результативно, чем использование метода покрытия условий
Вариант 2: метод покрытия операторов не позволяет обнаруживать ошибки в комбинированных логических условиях
Вариант 3: метод комбинаторного покрытия условий позволяет обнаружи-вать ошибки в комбинированных логических условиях
Вариант 4: тестирование программы методом покрытия условий более ре-зультативно, чем использование метода покрытия переходов
Вариант 5: тестирование программы методами черного ящика осуществля-ется без детализации внутренне структуры программы

104. Какое из сформулированных ниже утверждений верно?
Вариант 1: метод эквивалентных разбиений основывается на разбиении множества входных данных программы на непересекающиеся классы, обрабатываемые одинаковым образом
Вариант 2: метод эквивалентных разбиений основывается на разбиении множества входных данных программы на непрерывные под-диапазоны, обрабатываемые одинаковым образом
Вариант 3: метод эквивалентных разбиений основывается на разбиении множества правильных входных данных программы на непере-секающиеся классы, обрабатываемые одинаковым образом
Вариант 4: метод эквивалентных разбиений основывается на разбиении множества входных данных программы на подмножества дис-кретных значений, обрабатываемые одинаковым образом
Вариант 5: метод эквивалентных разбиений основывается на разбиении множества входных данных программы на подмножества кон-кретных значений, обрабатываемые одинаковым образом

105. Какое из сформулированных ниже утверждений верно?
Вариант 1: программная избыточность представляет собой введение допол-нительных модулей для контроля, диагностики и восстановле-ния работоспособности программной системы
Вариант 2: программная избыточность представляет собой введение допол-нительных ресурсов для дублированного хранения данных и кода во внешней памяти вычислительной системы
Вариант 3: программная избыточность представляет собой введение допол-нительных полей в обрабатываемые данные, позволяющих вос-станавливать искаженные значения
Вариант 4: программная избыточность представляет собой введение допол-нительных модулей для сигнализации и перезагрузки про-граммной системы
Вариант 5: программная избыточность представляет собой введение допол-нительных модулей для ускорения работы программы

106. Какое из сформулированных ниже утверждений верно?
Вариант 1: метод анализа граничных значений учитывает только правиль-ные значения
Вариант 2: метод анализа граничных значений не может применяться к дискретным наборам входных значений программы
Вариант 3: метод анализа граничных значений более эффективен, чем ме-тод эквивалентных разбиений
Вариант 4: метод анализа граничных значений позволяет снизить трудоем-кость тестирования программы за счет ограничения количества тестов
Вариант 5: метод анализа граничных значений позволяет тестировать про-граммы любой сложности

107. Какое из сформулированных ниже утверждений неверно?

Вариант 1: надежность программы зависит от количества комментариев в ее исходном тексте

Вариант 2: надежность программы зависит от объема глобальных данных, доступных функциям программы

Вариант 3: надежность программы зависит от числа ее модулей

Вариант 4: надежность программы зависит от числа функций в ее исходном тексте

Вариант 5: надежность программы зависит от ее объема

Задания в открытой форме

108. Какой тип межмодульного сцепления характерен для следующей функции?

```
Function MaxA: integer;
Var i:word;
begin
     MaxA: =a[Low(a)];
     for i: = Low (a) + l to High(a) do
     if a [i]>MaxA then MaxA: = a [i];
end;
```

109. Какой тип сцепления соответствует следующей функции лексического анализатора:

```
char CTokenManager::GetToken( token_class_t token_class, size_t to-
ken_index ) {
    char token;
    switch ( token_class ) {
        case tc_punctuator  : token = this->punctuators[ token_index
]; break;
        case tc_letter      : token = this->letters[ token_index ];
break;
        case tc_digit       : token = this->digits[ token_index ];
break;
        case tc_underscore  : token = this->underscore[ token_index
]; break;
    }
    return token;
}
```

110. Какие факторы определяют надежность программы?

111. Что ухудшает эффективность программы по времени выполнения?

112. Что ухудшает эффективность программы по объему памяти?

113. Нисходящее проектирование заключается в .....

114. Восходящее проектирование применяется для .....

115. Функциональная точка это .....

116. Что понимается под CASE-средствами?

117. Что является условием завершения нисходящего проектирования программной системы?

Задание на установление правильной последовательности

118. Установите последовательность жизненного цикла программного обеспечения:

период времени от момента зарождения идеи создания ПО до момента завершения его поддержки

119. Установите последовательность этапов разработки ПО?
анализ требований к программной системе, проектирование программной системы, анализ требований к компонентам си-стемы, проектирование компонент системы, программирова-ние компонент, компоновка и тестирование программной си-стемы, документирование

120. Расположите в порядке следования этапы жизненного цикла программы.
Тестирование, отладка, разработка, сопровождение.

Задание на установление соответствия:

121. Установите верное соответствие между выражениями
декомпозиция программной системы на модули позволяет организовать параллельную работу над несколькими модулями
декомпозиция программной системы по функциональному принципу более эффективна, чем декомпозиция по потокам данных

122. соответствие между языками программирования высокого и низкого уровня:
C++, Pascal, C#, Python, Basic, C, Assembler, машинные коды.

123. Установите верное соответствие между перечисленными ниже факторами и ресурсной эффективность программы?
емкостная сложность алгоритма решения задачи;
количество файлов в проекте программы;
правила выбора идентификаторов для программных сущностей;
количество и число итераций циклов программы;
число глобальных переменных в программе.

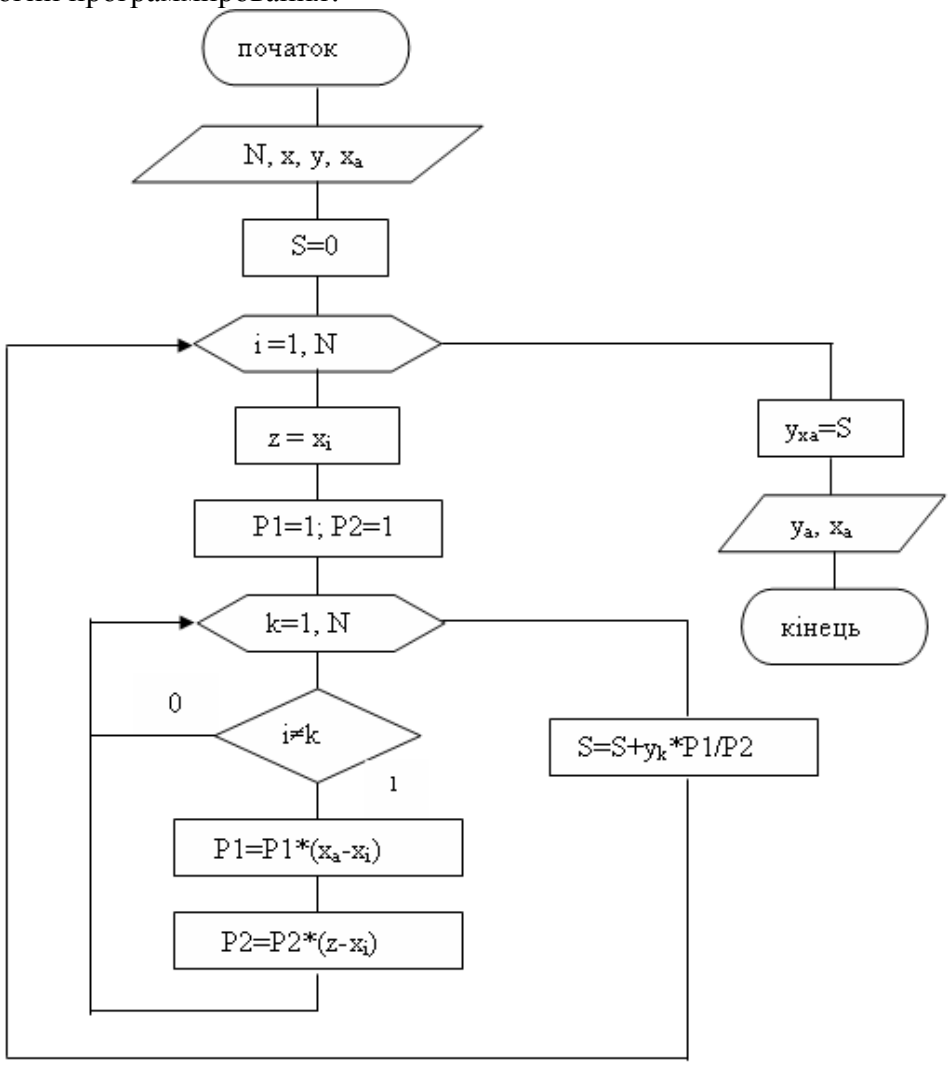
124. Установите верное соответствие между стилем программирования и определениями:
правила выбора идентификаторов и правила распределения и заполнения комментариев
только правила распределения и заполнения комментариев

125. Установите соответствие между интегрированными средами разработки ПО и
Инструментальным ПО, прикладным ПО, системным ПО

Компетентностно-ориентированные задачи

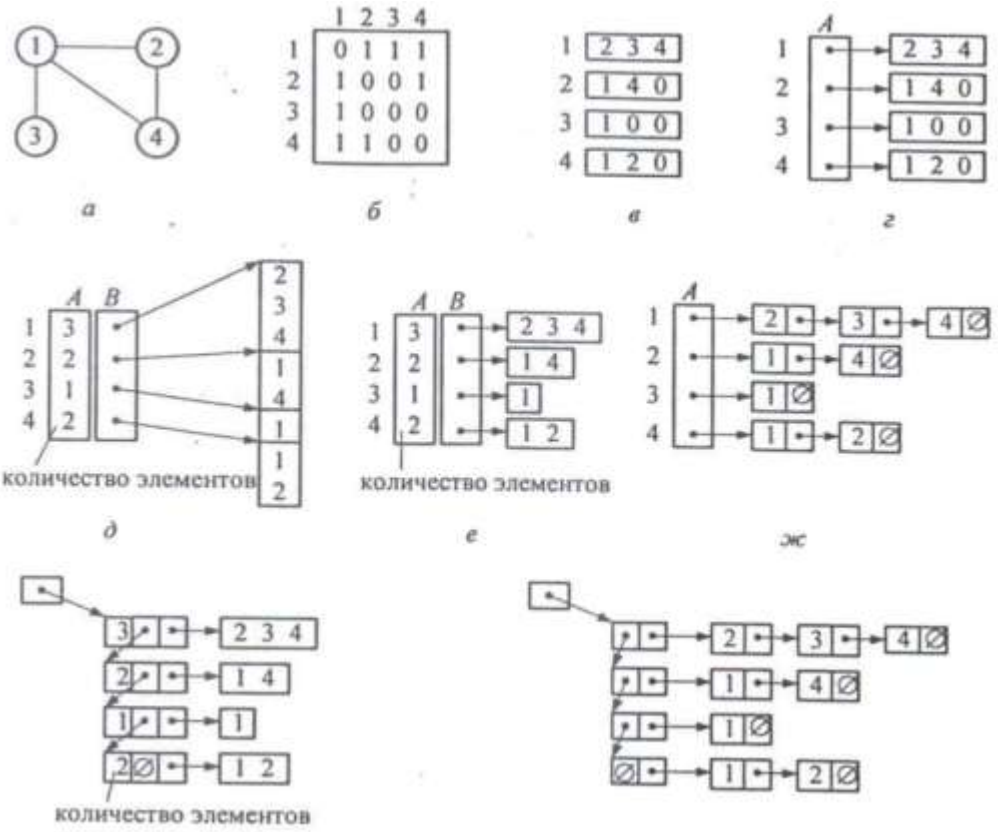
1. Разработать функцию, которая подсчитывает высоту (число уровней) заданного бинарного дерева, учитывая требования технологичности.
2. Разработать функцию, которая подсчитывает число листьев в заданном бинарном дереве, учитывая требования технологичности.
3. Разработать функцию, которая находит наибольшее значение в заданном бинарном дереве, учитывая требования технологичности.
4. Разработать функцию, которая находит номер уровня, на котором находится наибольшее значение в заданном бинарном дереве, учитывая требования технологичности.
5. Разработать функцию, которая определяет количество отрицательных значений в заданном бинарном дереве, учитывая требования технологичности.
6. Разработать функцию, которая находит сумму элементов заданного уровня бинарного дерева, учитывая требования технологичности.

7. Перечислите формальные недостатки алгоритма, изображенного на рисунке, с точки зрения технологии программирования?

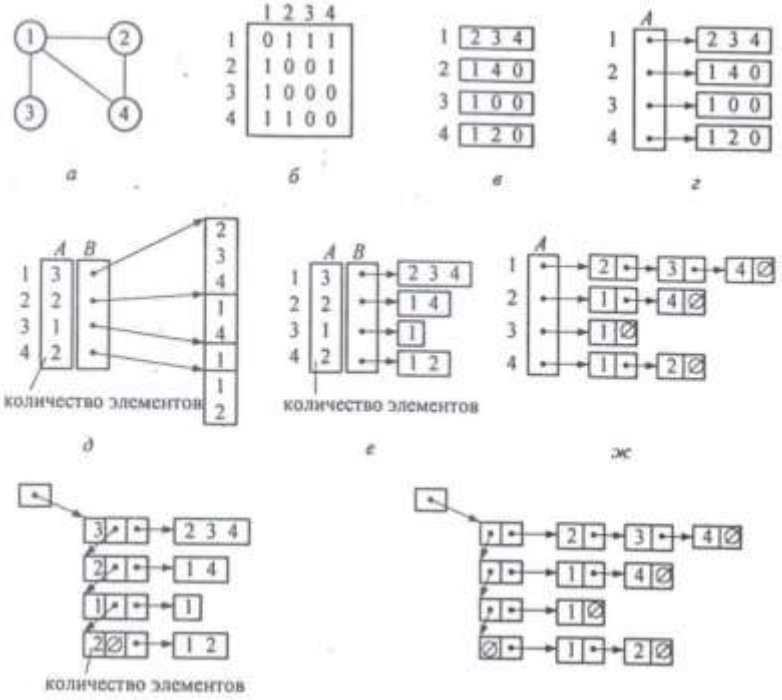


8. Разработать функцию, которая находит минимальный элемент среди элементов заданного уровня бинарного дерева, учитывая требования технологичности.

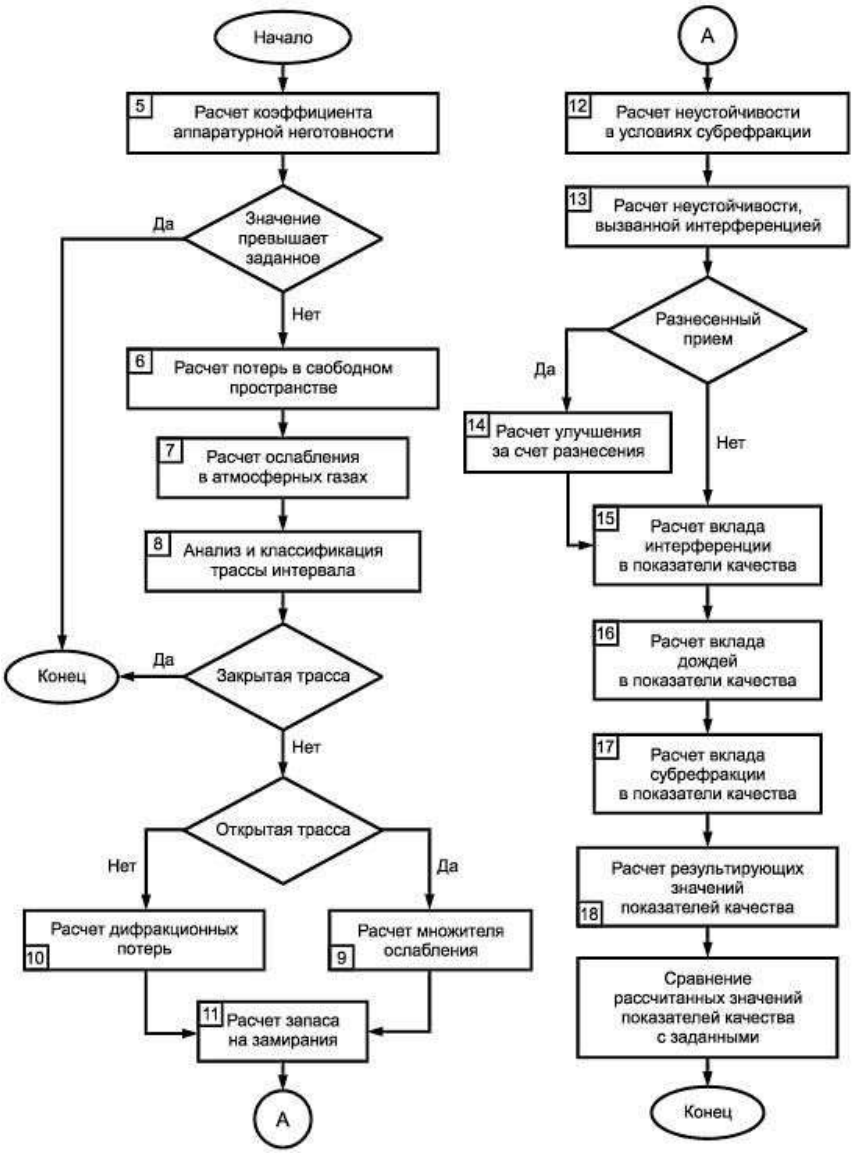
9. Какая из структур данных, приведенных на рисунке, наиболее эффективна по затратам памяти для представления графа?



10. Разработать функцию, которая сравнивает суммы элементов двух заданных уровней бинарного дерева, учитывая требования технологичности.
11. Какая из структур данных, приведенных на рисунке, наиболее эффективна по времени удаления произвольной вершины из графа?



12. Разработать функцию, которая находит число нулевых элементов на заданном уровне бинарного дерева, учитывая требования технологичности.
13. Перечислите формальные недостатки алгоритма, изображенного на рисунке, с точки зрения технологии программирования?



14. Разработать функцию, которая проверяет, есть ли хотя бы один лист на заданном уровне бинарного дерева, учитывая требования технологичности.
15. Разработать программу для подсчета количества повторений заданного числа в заданной прямоугольной матрице.
16. Разработать программу для определения номера строки заданной прямоугольной матрицы, имеющей наибольшую сумму элементов.
17. Разработать программу для сортировки строк заданной прямоугольной матрицы по убыванию элементов.
18. Написать программу для поиска столбца заданной прямоугольной матрицы, содержащего больше всего отрицательных элементов.
19. Разработать программу для транспонирования заданной квадратной матрицы.
20. Разработать программу для вычисления определенного интеграла методом трапеций.
21. Разработать класс student
Поля: ФИО, год рождения, пол, стипендия
функции: получить и записать значения полей, функция вывода значений полей на экран, запись в файл, чтение из файла.

Критерии оценки:

- задание в закрытой форме –2 балла,
- задание в открытой форме – 2 балла,
- задание на установление правильной последовательности – 2 балла,

- задание на установление соответствия – 2 балла,
- решение компетентностно-ориентированной задачи – 6 баллов.

Составитель В.С. Панищев

«31» августа 2022 г