МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«ИРКУТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ФГБОУ ВО «ИГУ»
Кафедра физико-химической биологии, биоинженерии и биоинформатики

ФОНД ОЦЕНОЧНЫХ МАТЕРИАЛОВ
для проведения текущего контроля и промежуточной аттестации по дисциплине:

Б1.О.18 **«Основы программирования»**

Специальность: 06.05.01 «Биоинженерия и биоинформатика»

Специализация: «Биоинженерия и биоинформатика»

Квалификация выпускника: биоинженер и биоинформатик

Форма обучения: очная с элементами электронного обучения и дистанционных образовательных технологий

Согласовано с УМК биолого-почвенного факультета
Протокол № 5 от 21 марта 2025 г.
Председатель _____ А.Н. Матвеев

Рекомендовано кафедрой физико-химической биологии, биоинженерии и биоинформатики
Протокол № 12 от 19 марта 2025 г.
Зав. кафедрой _____ В.П. Саловарова

Иркутск 2025 г.

ФОНД ОЦЕНОЧНЫХ МАТЕРИАЛОВ

Разработан для учебной дисциплины Б1.О.18 «ОСНОВЫ ПРОГРАММИРОВАНИЯ» 06.05.01 «Биоинженерия и биоинформатика», Специализация: «Биоинженерия и биоинформатика». Фонд оценочных материалов (ФОМ) включает оценочные материалы для проведения текущего контроля, промежуточной аттестации в форме экзамена.

Оценочные материалы соотнесены с требуемыми результатами освоения образовательной программы 06.05.01 «Биоинженерия и биоинформатика», в соответствии с содержанием рабочей программы учебной дисциплины Б1.О.18 «Основы программирования» с учетом ОПОП.

Нормативные документы, регламентирующие разработку ФОМ:
- статья 2, часть 9 Федерального закона «Об образовании в Российской Федерации», ФЗ-273, от 29.12.2012 г.;
- ФГОС ВО по специальности 06.05.01 «Биоинженерия и биоинформатика», утвержденный приказом Министерства науки и высшего образования Российской Федерации 12 августа 2020 г. № 973.

1.Компетенции, формируемые в процессе изучения дисциплины (1 курс, 1 семестр)

ОПК-6: Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения.

ОПК-7: Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности.

Компетенция	**Индикаторы компетенций**	**Результаты обучения**	**Формы и методы контроля и оценки**
ОПК-6 Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения	*ИДК ОПК-6.1* Знает принципы создания компьютерных программ, используемых в биоинформатике и биоинженерии	Знать: основные синтаксические конструкции и типы данных языков программирования С и С++ . Уметь: строить сложные алгоритмы с помощью принципов структурного и нисходящего программирования. Владеть: навыками полстроения сложных алгоритмов для обработки биологических данных, производить отладку и тестирование разработанных алгоритмов	**Текущий контроль**: - письменная работа (решение самостоятельных заданий) **Промежуточная аттестация**: экзамен
	ИДК ОПК-6.2 Использует современные IT-технологии при сборе, анализе, обработке и представлении информации	Знать: современные библиотеки и наборы функций для языков программирования С и С++, применяемы для анализа и визуализации сложных данных. Уметь: использовать различные библиотеки функции для анализа сложных данных при построении алгоритмов.	**Текущий контроль**: - письменная работа (решение самостоятельных заданий) **Промежуточная аттестация**: экзамен
	ИДК ОПК-6.3 Использует навыки создания	Знать: классификацию алгоритмов, основные типы алгоритмов, синтаксис базовых	**Текущий контроль**: - письменная работа

	компьютерных программ, баз данных и иные программных продуктов, используемых в биоинженерии и биоинформатике	алгоритмов в языках программирования С и С++. Уметь: анализировать входные и выходные данные разрабатываемого алгоритма. Владеть: навыками использования полученные знания для решения профессиональных задач	(решение самостоятельных заданий) **Промежуточная аттестация**: экзамен
ОПК-7 Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности	*ИДК ОПК-7.1* Демонстрирует теоретические и практические навыки использования современных информационных технологий в области профессианальной деятельности	Знать: основные понятие и термины, применяемы при описании и разработки алгоритмов на языках программирования С и С++. Уметь: использовать полученные знания для поиска готовых алгоритмических решений различных задач в сети интернет и научной литературе. Владеть: навыками использования готовых алгоритмов при построения собственных решений для анализа сложных данных	**Текущий контроль**: - письменная работа (решение самостоятельных заданий) **Промежуточная аттестация**: экзамен
	ИДК ОПК-7.1 Использует современные информационные технологии в рамках освоения материала и реализации задач в области профессиональной деятельности	Знать: спектр современных средств языков программирования С и С++ для решения задач в рамках современные информационные технологии Уметь: использовать полученные знания и навыки для решения профессиональных задач Владеть: теоретическими званиями и практическими навыками в области профессиональной деятельности.	**Текущий контроль**: - письменная работа (решение самостоятельных заданий) **Промежуточная аттестация**: экзамен

2. Оценочные материалы текущего контроля

В рамках дисциплины «Основы программирования» используются следующие формы текущего контроля - письменная работа по решению самостоятельных заданий (все формулировки заданий для самостоятельного решения с необходимыми сопроводительными материалами выложены на образовательном портале ИГУ в темах курса «Основы программирования»);

Перечень письменных работ для самостоятельного выполнения по разделам – темам дисциплины.

Задание по теме 1:
Задание 1: "Привет, мир!" с пользовательским вводом

- **Задача:** Напишите программу, которая запрашивает у пользователя его имя и затем выводит приветствие, используя введенное имя. Например: "Здравствуйте, [имя]!".
- **Указания:**
  - Используйте `iostream` для ввода-вывода (в C++).
  - Используйте `stdio.h` и `scanf/printf` для ввода-вывода (в C).
  - Используйте `std::string` для хранения имени (в C++) или `char[]` (в C).

Ответ:
Код в C++:

```
#include <iostream>
#include <string>

int main() {
  std::string name;

  std::cout << "Пожалуйста, введите ваше имя: ";
  std::cin >> name;

  std::cout << "Здравствуйте, " << name << "!" << std::endl;

  return 0;
}
```

Код в C:

```
#include <stdio.h>

int main() {
  char name[50]; // Выделяем место для имени (до 49 символов + '\0')

  printf("Пожалуйста, введите ваше имя: ");
  scanf("%49s", name); // Ограничение длины во избежание переполнения буфера

  printf("Здравствуйте, %s!\n", name);

  return 0;
}
```

Задание по теме 2:
Калькулятор простых арифметических операций
Задание: Базовый арифметический калькулятор (линейный алгоритм)
Описание:
Напишите консольную программу на C или C++, которая выполняет следующие действия:

1. **Запрашивает у пользователя два числа:** Программа должна запросить у пользователя два числа с плавающей точкой (тип `double`).

2. **Выполняет арифметические операции:** Программа должна выполнить следующие операции над введенными числами:
   - Сложение
   - Вычитание
   - Умножение
   - Деление
3. **Выводит результаты:** Программа должна вывести результаты всех четырех операций на экран, каждое на отдельной строке, с поясняющим текстом.

Указания:

- Используйте `iostream` для ввода-вывода (в C++) или `stdio.h` для ввода-вывода (в C).
- Используйте тип данных `double` для чисел, чтобы обеспечить точность.
- Убедитесь, что программа компилируется и запускается без ошибок.
- Не требуется обработка ошибок (например, деления на ноль) в этом задании. Это может быть заданием на повышение сложности.
  -

Ответ:

Код в C++:

```
#include <iostream>
#include <iomanip> // Для форматирования вывода

int main() {
  double num1, num2;

  std::cout << "Введите первое число: ";
  std::cin >> num1;

  std::cout << "Введите второе число: ";
  std::cin >> num2;

  double sum = num1 + num2;
  double difference = num1 - num2;
  double product = num1 * num2;
  double quotient = num1 / num2;

  std::cout << "Сумма: " << std::fixed << std::setprecision(2) << sum << std::endl;
  std::cout << "Разность: " << std::fixed << std::setprecision(2) << difference << std::endl;
  std::cout << "Произведение: " << std::fixed << std::setprecision(2) << product << std::endl;
  std::cout << "Частное: " << std::fixed << std::setprecision(2) << quotient << std::endl;

  return 0;
}
```

Код в C:

```
#include <stdio.h>

int main() {
  double num1, num2;
  double sum, difference, product, quotient;

  printf("Введите первое число: ");
  scanf("%lf", &num1);

  printf("Введите второе число: ");
  scanf("%lf", &num2);
```

```
  sum = num1 + num2;
  difference = num1 - num2;
  product = num1 * num2;
  quotient = num1 / num2;

  printf("Сумма: %.2lf\n", sum);
  printf("Разность: %.2lf\n", difference);
  printf("Произведение: %.2lf\n", product);
  printf("Частное: %.2lf\n", quotient);

  return 0;
}
```

Задание по теме 3:

Задание: Арифметический калькулятор

Описание:

Напишите консольную программу на C или C++, которая выполняет следующие действия:

1. **Запрашивает у пользователя два числа:** Программа должна предложить пользователю ввести два числа с плавающей точкой (например, `double` в C++ или `double` в C).
2. **Запрашивает операцию:** Программа должна предложить пользователю выбрать арифметическую операцию для выполнения (+, -, *, /).
3. **Выполняет операцию:** В зависимости от выбранной операции, программа должна выполнить сложение, вычитание, умножение или деление введенных чисел.
4. **Выводит результат:** Программа должна вывести результат выполненной операции на экран.
5. **Обработка ошибок:** Программа должна корректно обрабатывать случай деления на ноль, выводя сообщение об ошибке, а не приводя к аварийному завершению.
6. **Повторение (бонус):** Добавьте возможность зациклить выполнение программы, чтобы пользователь мог выполнять несколько операций подряд, пока не захочет завершить её.

Указания:

- Используйте `iostream` для ввода-вывода (в C++) или `stdio.h` для ввода-вывода (в C).
- Используйте условные операторы (`if`, `else if`, `else` или `switch`) для выбора операции.
- Для хранения чисел используйте тип `double`, чтобы обеспечить точность вычислений.
- Обратите внимание на обработку случая деления на ноль.

Ответ:

Код в C++:

```
#include <iostream>
#include <iomanip> // Для управления точностью вывода

int main() {
  double num1, num2;
  char operation;
  bool repeat = true; // Для повторения

  while (repeat) {
    std::cout << "Введите первое число: ";
    std::cin >> num1;

    std::cout << "Введите второе число: ";
    std::cin >> num2;

    std::cout << "Введите операцию (+, -, *, /): ";
    std::cin >> operation;
```

```
    double result;

    switch (operation) {
      case '+':
        result = num1 + num2;
        break;
      case '-':
        result = num1 - num2;
        break;
      case '*':
        result = num1 * num2;
        break;
      case '/':
        if (num2 == 0) {
          std::cout << "Ошибка: Деление на ноль!" << std::endl;
          continue; // Переход к следующей итерации цикла
        }
        result = num1 / num2;
        break;
      default:
        std::cout << "Ошибка: Недопустимая операция!" << std::endl;
        continue; // Переход к следующей итерации цикла
    }

    std::cout << "Результат: " << std::fixed << std::setprecision(2) << result << std::endl; // Вывод с фиксированным числом знаков после запятой

    char choice;
    std::cout << "Выполнить еще одну операцию? (y/n): ";
    std::cin >> choice;

    if (choice != 'y') {
      repeat = false;
    }
  }

  std::cout << "Программа завершена." << std::endl;

  return 0;
}
```

Код в C:

```
#include <stdio.h>

int main() {
  double num1, num2;
  char operation;
  char choice;
  int repeat = 1; // Для повторения

  while (repeat) {
```

```
    printf("Введите первое число: ");
    scanf("%lf", &num1);

    printf("Введите второе число: ");
    scanf("%lf", &num2);

    printf("Введите операцию (+, -, *, /): ");
    scanf(" %c", &operation); // Обратите внимание на пробел перед %c для пропуска пробельных символов

    double result;

    switch (operation) {
      case '+':
        result = num1 + num2;
        break;
      case '-':
        result = num1 - num2;
        break;
      case '*':
        result = num1 * num2;
        break;
      case '/':
        if (num2 == 0) {
          printf("Ошибка: Деление на ноль!\n");
          continue; // переход к следующей итерации цикла
        }
        result = num1 / num2;
        break;
      default:
        printf("Ошибка: Недопустимая операция!\n");
        continue; // переход к следующей итерации цикла
    }

    printf("Результат: %.2lf\n", result); // Вывод с двумя знаками после запятой

    printf("Выполнить еще одну операцию? (y/n): ");
    scanf(" %c", &choice);

    if (choice != 'y') {
      repeat = 0;
    }
  }

  printf("Программа завершена.\n");

  return 0;
}
```

Задание по теме 4:

Напишите программу на C++, которая вычисляет факториал числа, введенного пользователем с клавиатуры. В отличие от предыдущего задания, все три варианта

вычисления (с использованием циклов `for`, `while` и `do...while`) должны быть реализованы *внутри* функции `main`, без создания отдельных функций для каждого цикла.
Программа должна:

1. Запросить у пользователя целое неотрицательное число.
2. Проверить ввод: Если пользователь ввел отрицательное число, программа должна вывести сообщение об ошибке и предложить ввести число еще раз.
3. Вычислить факториал введенного числа тремя разными способами (с использованием циклов `for`, `while` и `do...while`) непосредственно в `main`.
4. Вывести результаты, полученные каждым циклом, на экран, сопроводив их поясняющим текстом.

Указания:

- Используйте `iostream` для ввода-вывода (в C++) или `stdio.h` для ввода-вывода (в C).
- Используйте тип данных `long long int` для хранения факториала, чтобы избежать переполнения.
- Не забудьте про случай, когда n = 0 (факториал 0 равен 1).
- Постарайтесь сделать код читаемым, несмотря на то, что все вычисления находятся в `main`.

Ответ:

```
#include <iostream>

int main() {
  int number;

  while (true) {
    std::cout << "Введите целое неотрицательное число: ";
    std::cin >> number;

    if (number < 0) {
      std::cout << "Ошибка: Введите неотрицательное число!" << std::endl;
    } else {
      break; // Выход из цикла, если ввод корректен
    }
  }

  // Факториал с помощью цикла for
  long long int factorial_for = 1;
  for (int i = 1; i <= number; ++i) {
    factorial_for *= i;
  }

  // Факториал с помощью цикла while
  long long int factorial_while = 1;
  int i = 1;
  while (i <= number) {
    factorial_while *= i;
    ++i;
  }

  // Факториал с помощью цикла do...while
  long long int factorial_do_while = 1;
  i = 1;
  if (number == 0) {
       factorial_do_while = 1;
    } else {
```

```
    do {
        factorial_do_while *= i;
        ++i;
    } while (i <= number);
}


 std::cout << "Факториал (" << number << ") (for): " << factorial_for << std::endl;
 std::cout << "Факториал (" << number << ") (while): " << factorial_while << std::endl;
 std::cout << "Факториал (" << number << ") (do...while): " << factorial_do_while << std::endl;

 return 0;
}
```

Задание по теме 5:
Напишите программу на C++, которая вычисляет среднее значение и стандартное отклонение для массива из 12 элементов с плавающей точкой, введенных пользователем с клавиатуры. Все вычисления и ввод/вывод должны выполняться непосредственно в функции `main`, без создания отдельных функций.
Программа должна:

1. Создать массив из 12 элементов типа `double`.
2. Запросить у пользователя ввод 12 чисел с плавающей точкой, которые будут сохранены в массиве.
3. Вычислить среднее значение (average) элементов массива.
4. Вычислить стандартное отклонение (standard deviation) элементов массива.
5. Вывести на экран среднее значение и стандартное отклонение с поясняющими текстовыми сообщениями. Вывод значений должен быть с двумя знаками после запятой.

Формулы:

- **Среднее значение (average):** `average = (sum of all elements) / (number of elements)`
- **Стандартное отклонение (standard deviation):**
  1. Вычислить разницу между каждым элементом и средним значением.
  2. Возвести каждую разницу в квадрат.
  3. Найти среднее значение этих квадратов разностей (дисперсию).
  4. Взять квадратный корень из дисперсии.

  `standard deviation = sqrt( (sum of (x_i - average)^2) / (number of elements) )`

Указания:

- Используйте `iostream` для ввода-вывода (в C++) или `stdio.h` для ввода-вывода (в C).
- Используйте тип данных `double` для массива и всех вычислений.
- Используйте библиотеку `<cmath>` (C++) или `<math.h>` (C) для функции `sqrt()` (квадратный корень).
- Постарайтесь сделать код максимально читаемым, используя понятные имена переменных и комментарии.

Ответ:

```
#include <iostream>
#include <cmath>
#include <iomanip> // Для форматирования вывода

int main() {
 const int SIZE = 12;
 double numbers[SIZE];
 double sum = 0.0;
 double average, standardDeviation;
```

```
 // Шаг 1: Ввод элементов массива
 std::cout << "Введите 12 чисел с плавающей точкой:" << std::endl;
 for (int i = 0; i < SIZE; ++i) {
  std::cout << "Число " << i + 1 << ": ";
  std::cin >> numbers[i];
  sum += numbers[i]; // Сразу считаем сумму для среднего значения
 }

 // Шаг 2: Вычисление среднего значения
 average = sum / SIZE;

 // Шаг 3: Вычисление стандартного отклонения
 double sumOfSquaredDifferences = 0.0;
 for (int i = 0; i < SIZE; ++i) {
  sumOfSquaredDifferences += pow(numbers[i] - average, 2);
 }
 standardDeviation = sqrt(sumOfSquaredDifferences / SIZE);

 // Шаг 4: Вывод результатов
 std::cout << std::fixed << std::setprecision(2); // Устанавливаем формат вывода
 std::cout << "Среднее значение: " << average << std::endl;
 std::cout << "Стандартное отклонение: " << standardDeviation << std::endl;

 return 0;
}
```

Задание по теме 6:

Напишите программу на C или C++ которая:

1. Запрашивает у пользователя желаемое количество элементов массива (N).
2. Создает массив из N элементов типа `double`. Если N <=0 программа завершается с выводом соответствующего сообщения.
3. Запрашивает у пользователя ввод N чисел с плавающей точкой, которые будут сохранены в массиве.
4. Вычислит среднее значение (average) и стандартное отклонение (standard deviation) элементов массива.
5. Находит максимальный и минимальный элементы массива, выводит на экран их индексы и значения.
6. Сортирует массив в порядке возрастания (например, с помощью алгоритма "пузырьковой сортировки").
7. Выводит отсортированный массив на экран.
8. Вывести на экран среднее значение и стандартное отклонение с поясняющими текстовыми сообщениями. Вывод значений должен быть с двумя знаками после запятой.

Вся функциональность должна быть реализована ВНУТРИ функции `main`, без создания дополнительных функций.

Формулы:

- **Среднее значение (average):** `average = (sum of all elements) / (number of elements)`
- **Стандартное отклонение (standard deviation):** `standard deviation = sqrt( (sum of (x_i - average)^2) / (number of elements) )`

Указания:

- Используйте `iostream` для ввода-вывода (в C++) или `stdio.h` для ввода-вывода (в C).
- Используйте тип данных `double` для массива и всех вычислений.
- Используйте библиотеку `<cmath>` (C++) или `<math.h>` (C) для функции `sqrt()` и `pow()`.
- Реализуйте алгоритм сортировки "пузырьком" непосредственно в `main`.

- Обязательно проверьте, чтобы N было положительным числом. Если N <= 0, выведите сообщение об ошибке и завершите программу.
- Постарайтесь придерживаться хорошего стиля кодирования, чтобы код оставался читаемым.

Ответ:

```
#include <iostream>
#include <cmath>
#include <iomanip>
#include <limits> // Для numeric_limits

int main() {
   int n;

   // Шаг 1: Ввод размера массива
   std::cout << "Введите количество элементов массива (N): ";
   std::cin >> n;

   if (n <= 0) {
      std::cerr << "Ошибка: Размер массива должен быть положительным числом." << std::endl;
      return 1; // Выход с кодом ошибки
   }

   // Шаг 2: Создание массива
   double* numbers = new double[n]; // Динамическое выделение памяти

   // Шаг 3: Ввод элементов массива
   std::cout << "Введите " << n << " чисел с плавающей точкой:" << std::endl;
   double sum = 0.0;
   for (int i = 0; i < n; ++i) {
      std::cout << "Число " << i + 1 << ": ";
      std::cin >> numbers[i];

      // Проверка на некорректный ввод
      if (std::cin.fail()) {
         std::cerr << "Ошибка: Некорректный ввод. Пожалуйста, введите число." << std::endl;
         std::cin.clear();
         std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
         --i; // Повторяем ввод текущего элемента
         continue;
      }
      sum += numbers[i];
   }

   // Шаг 4: Вычисление среднего значения
   double average = sum / n;

   // Шаг 5: Вычисление стандартного отклонения
   double sumOfSquaredDifferences = 0.0;
   for (int i = 0; i < n; ++i) {
      sumOfSquaredDifferences += pow(numbers[i] - average, 2);
```

```
    }
    double standardDeviation = sqrt(sumOfSquaredDifferences / n);

    // Шаг 6: Поиск максимального и минимального элементов
    int minIndex = 0, maxIndex = 0;
    for (int i = 1; i < n; ++i) {
        if (numbers[i] < numbers[minIndex]) {
            minIndex = i;
        }
        if (numbers[i] > numbers[maxIndex]) {
            maxIndex = i;
        }
    }

    std::cout << "Минимальный элемент: numbers[" << minIndex << "] = " << numbers[minIndex] << std::endl;
    std::cout << "Максимальный элемент: numbers[" << maxIndex << "] = " << numbers[maxIndex] << std::endl;


    // Шаг 7: Сортировка пузырьком
    for (int i = 0; i < n - 1; ++i) {
        for (int j = 0; j < n - i - 1; ++j) {
            if (numbers[j] > numbers[j + 1]) {
                // Обмен элементов
                double temp = numbers[j];
                numbers[j] = numbers[j + 1];
                numbers[j + 1] = temp;
            }
        }
    }

    // Шаг 8: Вывод отсортированного массива
    std::cout << "Отсортированный массив:" << std::endl;
    for (int i = 0; i < n; ++i) {
        std::cout << numbers[i] << " ";
    }
    std::cout << std::endl;

    // Шаг 9: Вывод среднего и стандартного отклонения
    std::cout << std::fixed << std::setprecision(2);
    std::cout << "Среднее значение: " << average << std::endl;
    std::cout << "Стандартное отклонение: " << standardDeviation << std::endl;

    delete[] numbers; // Освобождаем выделенную память
    return 0;
}
```

Задание по теме 7:

Напишите программу на C++, которая демонстрирует работу с указателями и массивами. Программа должна:

1. **Объявить массив**: Объявите массив целых чисел `int` размером 10 (статический массив).

2. **Инициализировать массив**: Инициализируйте массив случайными числами в диапазоне от 1 до 100 (включительно). Используйте `rand()` и `srand()` для генерации случайных чисел.
3. **Вывод массива**: Выведите элементы массива на экран, используя указатель и арифметику указателей. Каждый элемент должен быть выведен на новой строке с указанием его индекса.
4. **Обратный порядок**: Выведите элементы массива на экран в обратном порядке, *также* используя указатель и арифметику указателей. Каждый элемент должен быть выведен на новой строке с указанием его индекса.
5. **Поиск максимума и минимума**: Найдите минимальный и максимальный элементы массива, используя указатели для доступа к элементам массива. Выведите значения минимального и максимального элементов, а также их индексы.
6. **Обращение порядка элементов**: Измените порядок элементов в массиве на обратный, используя указатели для обмена значениями. Не используйте дополнительный массив. То есть, нужно поменять местами первый и последний элементы, второй и предпоследний, и т.д.
7. **Вывод после обращения**: Выведите элементы массива на экран *после* обращения порядка элементов, используя указатель (как в шаге 3).

Все операции с массивом должны выполняться с использованием указателей, а не индексов (за исключением, возможно, инициализации массива). Вся логика должна быть реализована непосредственно в функции `main` без создания дополнительных функций.

Указания:

- Используйте `iostream` (C++) или `stdio.h` (C) для ввода-вывода.
- Используйте `<cstdlib>` (C++) или `stdlib.h` (C) для функций `rand()` и `srand()`.
- Используйте `<ctime>` (C++) или `time.h` (C) для функции `time()` (для инициализации генератора случайных чисел).
- Обратите внимание на арифметику указателей. `*(ptr + i)` эквивалентно `array[i]`, если `ptr` указывает на начало массива `array`.

Ответ:

```
#include <iostream>
#include <cstdlib> // Для rand() и srand()
#include <ctime>   // Для time()

int main() {
   const int SIZE = 10;
   int numbers[SIZE];
   int* ptr = numbers; // Указатель на первый элемент массива

   // 1. Инициализация массива случайными числами
   srand(time(0)); // Инициализация генератора случайных чисел
   for (int i = 0; i < SIZE; ++i) {
      numbers[i] = rand() % 100 + 1; // Случайные числа от 1 до 100
   }

   // 2. Вывод массива с использованием указателя
   std::cout << "Массив:" << std::endl;
   for (int i = 0; i < SIZE; ++i) {
      std::cout << "numbers[" << i << "] = " << *(ptr + i) << std::endl;
   }
   std::cout << std::endl;

   // 3. Вывод массива в обратном порядке с использованием указателя
   std::cout << "Массив в обратном порядке:" << std::endl;
   for (int i = SIZE - 1; i >= 0; --i) {
      std::cout << "numbers[" << i << "] = " << *(ptr + i) << std::endl;
```

```
    }
    std::cout << std::endl;

    // 4. Поиск максимума и минимума с использованием указателей
    int* minPtr = ptr;
    int* maxPtr = ptr;
    for (int i = 1; i < SIZE; ++i) {
        if (*(ptr + i) < *minPtr) {
            minPtr = ptr + i;
        }
        if (*(ptr + i) > *maxPtr) {
            maxPtr = ptr + i;
        }
    }

    std::cout << "Минимальный элемент: numbers[" << (minPtr - numbers) << "] = " << *minPtr << std::endl;
    std::cout << "Максимальный элемент: numbers[" << (maxPtr - numbers) << "] = " << *maxPtr << std::endl;
    std::cout << std::endl;

    // 5. Обращение порядка элементов в массиве с использованием указателей
    for (int i = 0; i < SIZE / 2; ++i) {
        int temp = *(ptr + i);
        *(ptr + i) = *(ptr + SIZE - 1 - i);
        *(ptr + SIZE - 1 - i) = temp;
    }

    // 6. Вывод массива после обращения
    std::cout << "Массив после обращения порядка элементов:" << std::endl;
    for (int i = 0; i < SIZE; ++i) {
        std::cout << "numbers[" << i << "] = " << *(ptr + i) << std::endl;
    }

    return 0;
}
```

Задание по теме 8:

Напишите программу на C или C++, которая:

1. Запрашивает у пользователя желаемое количество элементов массива (N).
2. Создает массив из N элементов типа `double`. Если N <=0 программа завершается с выводом соответствующего сообщения.
3. Запрашивает у пользователя ввод N чисел с плавающей точкой, которые будут сохранены в массиве.
4. **Вычисляет среднее значение**: Реализуйте функцию `double calculateAverage(double arr[], int n)`, которая принимает массив `arr` и его размер `n` в качестве аргументов и возвращает среднее значение элементов массива.
5. **Вычисляет стандартное отклонение**: Реализуйте функцию `double calculateStandardDeviation(double arr[], int n, double average)`, которая принимает массив `arr`, его размер `n` и *уже вычисленное* среднее значение `average` в качестве аргументов и возвращает стандартное отклонение элементов массива.
6. Выводит на экран среднее значение и стандартное отклонение, полученные из соответствующих функций. Вывод значений должен быть с двумя знаками после запятой.

Формулы:

- **Среднее значение (average):** `average = (sum of all elements) / (number of elements)`
- **Стандартное отклонение (standard deviation):** `standard deviation = sqrt( (sum of (x_i - average)^2) / (number of elements) )`

Указания:

- Используйте `iostream` для ввода-вывода (в C++) или `stdio.h` для ввода-вывода (в C).
- Используйте тип данных `double` для массива и всех вычислений.
- Используйте библиотеку `<cmath>` (C++) или `<math.h>` (C) для функции `sqrt()` и `pow()`.
- Обязательно проверьте, чтобы N было положительным числом. Если N <= 0, выведите сообщение об ошибке и завершите программу.
- Постарайтесь придерживаться хорошего стиля кодирования, чтобы код оставался читаемым.
- Следуйте прототипам функций, указанным в описании.

Ответ:

```
#include <iostream>
#include <cmath>
#include <iomanip>
#include <limits> // Для numeric_limits

// Функция для вычисления среднего значения
double calculateAverage(double arr[], int n) {
    double sum = 0.0;
    for (int i = 0; i < n; ++i) {
        sum += arr[i];
    }
    return sum / n;
}

// Функция для вычисления стандартного отклонения
double calculateStandardDeviation(double arr[], int n, double average) {
    double sumOfSquaredDifferences = 0.0;
    for (int i = 0; i < n; ++i) {
        sumOfSquaredDifferences += pow(arr[i] - average, 2);
    }
    return sqrt(sumOfSquaredDifferences / n);
}

int main() {
    int n;

    // Шаг 1: Ввод размера массива
    std::cout << "Введите количество элементов массива (N): ";
    std::cin >> n;

    if (n <= 0) {
        std::cerr << "Ошибка: Размер массива должен быть положительным числом." <<
std::endl;
        return 1; // Выход с кодом ошибки
    }

    // Шаг 2: Создание массива
    double* numbers = new double[n]; // Динамическое выделение памяти
```

```
    // Шаг 3: Ввод элементов массива
    std::cout << "Введите " << n << " чисел с плавающей точкой:" << std::endl;
    for (int i = 0; i < n; ++i) {
        std::cout << "Число " << i + 1 << ": ";
        std::cin >> numbers[i];

        // Проверка на некорректный ввод
        if (std::cin.fail()) {
            std::cerr << "Ошибка: Некорректный ввод. Пожалуйста, введите число." << std::endl;
            std::cin.clear();
            std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
            --i; // Повторяем ввод текущего элемента
            continue;
        }
    }

    // Шаг 4: Вычисление среднего значения с использованием функции
    double average = calculateAverage(numbers, n);

    // Шаг 5: Вычисление стандартного отклонения с использованием функции
    double standardDeviation = calculateStandardDeviation(numbers, n, average);

    // Шаг 6: Вывод результата
    std::cout << std::fixed << std::setprecision(2);
    std::cout << "Среднее значение: " << average << std::endl;
    std::cout << "Стандартное отклонение: " << standardDeviation << std::endl;

    delete[] numbers; // Освобождаем выделенную память
    return 0;
}
```

Задание по теме 9:

Напишите программу на C++, которая выполняет следующие действия:

1. **Чтение из файла**:
   - Программа должна запросить у пользователя имя входного файла.
   - Программа должна прочитать размер массива `n` (целое число) из *первой строки* файла.
   - Программа должна читать `n` чисел с плавающей точкой типа `double` из последующих строк файла и сохранить их в массиве.
   - Если при чтении файла возникают ошибки (например, файл не существует, файл содержит некорректные данные или недостаточно чисел), программа должна вывести сообщение об ошибке и завершиться.
2. **Вычисление статистики**:
   - Реализуйте функцию `double calculateAverage(double arr[], int n)`, которая принимает массив `arr` и его размер `n` в качестве аргументов и возвращает среднее значение элементов массива.
   - Реализуйте функцию `double calculateStandardDeviation(double arr[], int n, double average)`, которая принимает массив `arr`, его размер `n` и *уже вычисленное* среднее значение `average` в качестве аргументов и возвращает стандартное отклонение элементов массива.

- Выведите на экран среднее значение и стандартное отклонение, полученные из соответствующих функций. Вывод значений должен быть с двумя знаками после запятой.

3. **Сортировка и запись в файл**:
   - Реализуйте функцию `void sortArray(double arr[], int n)`, которая сортирует массив `arr` по *возрастанию*. Можно использовать любой алгоритм сортировки (например, пузырьковую сортировку, сортировку вставками, сортировку выбором).
   - Программа должна запросить у пользователя имя выходного файла.
   - Запишите отсортированный массив в указанный файл. Каждое число должно быть записано на отдельной строке.

Формулы:

- **Среднее значение (average):** `average = (sum of all elements) / (number of elements)`
- **Стандартное отклонение (standard deviation):** `standard deviation = sqrt( (sum of (x_i - average)^2) / (number of elements) )`

Указания:

- Используйте `iostream` и `fstream` для работы с файлами (в C++) или `stdio.h` для работы с файлами (в C).
- Используйте тип данных `double` для массива и всех вычислений.
- Используйте библиотеку `<cmath>` (C++) или `<math.h>` (C) для функции `sqrt()` и `pow()`.
- Функции `calculateAverage` и `calculateStandardDeviation` должны быть реализованы так же, как и в предыдущем задании.
- Обязательно проверяйте успешность открытия и чтения файлов.
- Обязательно освобождайте выделенную память после использования.
- Придерживайтесь хорошего стиля кодирования.

Пример содержимого входного файла (`data.txt`):

```
5
1.0
2.5
3.7
4.2
5.9
```

Ответ:

```
#include <iostream>
#include <fstream>
#include <cmath>
#include <iomanip>
#include <string>
#include <vector>
#include <algorithm>

// Функция для вычисления среднего значения
double calculateAverage(double arr[], int n) {
   double sum = 0.0;
   for (int i = 0; i < n; ++i) {
      sum += arr[i];
   }
   return sum / n;
}

// Функция для вычисления стандартного отклонения
double calculateStandardDeviation(double arr[], int n, double average) {
   double sumOfSquaredDifferences = 0.0;
   for (int i = 0; i < n; ++i) {
      sumOfSquaredDifferences += pow(arr[i] - average, 2);
   }
```

```
    return sqrt(sumOfSquaredDifferences / n);
}

// Функция для сортировки массива (используем std::sort)
void sortArray(double arr[], int n) {
    std::sort(arr, arr + n);  // Используем стандартную функцию сортировки
}

int main() {
    std::string inputFileName, outputFileName;
    int n;
    double* numbers = nullptr;

    // Шаг 1: Получение имени входного файла
    std::cout << "Введите имя входного файла: ";
    std::cin >> inputFileName;

    // Открываем входной файл
    std::ifstream inputFile(inputFileName);
    if (!inputFile.is_open()) {
        std::cerr << "Ошибка: Не удалось открыть файл " << inputFileName << std::endl;
        return 1;
    }

    // Читаем размер массива из файла
    if (!(inputFile >> n)) {
        std::cerr << "Ошибка: Не удалось прочитать размер массива из файла." << std::endl;
        inputFile.close();
        return 1;
    }

    if (n <= 0) {
       std::cerr << "Ошибка: Некорректный размер массива (N <= 0)." << std::endl;
       inputFile.close();
       return 1;
    }

    // Выделяем память для массива
    numbers = new double[n];

    // Читаем элементы массива из файла
    for (int i = 0; i < n; ++i) {
        if (!(inputFile >> numbers[i])) {
            std::cerr << "Ошибка: Не удалось прочитать элемент " << i + 1 << " из файла." <<
std::endl;
            inputFile.close();
            delete[] numbers;
            return 1;
        }
    }

    // Закрываем входной файл
```

```
    inputFile.close();

    // Шаг 2: Вычисление среднего значения и стандартного отклонения
    double average = calculateAverage(numbers, n);
    double standardDeviation = calculateStandardDeviation(numbers, n, average);

    std::cout << std::fixed << std::setprecision(2);
    std::cout << "Среднее значение: " << average << std::endl;
    std::cout << "Стандартное отклонение: " << standardDeviation << std::endl;

    // Шаг 3: Получение имени выходного файла
    std::cout << "Введите имя выходного файла: ";
    std::cin >> outputFileName;

    // Сортируем массив
    sortArray(numbers, n);

    // Открываем выходной файл
    std::ofstream outputFile(outputFileName);
    if (!outputFile.is_open()) {
        std::cerr << "Ошибка: Не удалось открыть файл " << outputFileName << " для записи."
<< std::endl;
        delete[] numbers;
        return 1;
    }

    // Записываем отсортированный массив в файл
    for (int i = 0; i < n; ++i) {
        outputFile << numbers[i] << std::endl;
    }

    // Закрываем выходной файл
    outputFile.close();

    std::cout << "Отсортированный массив записан в файл " << outputFileName << std::endl;

    // Освобождаем память
    delete[] numbers;

    return 0;
}
```

Задание по теме 10:

Напишите программу на C++, которая выполняет следующие действия, **используя структуру** для хранения данных:

1. **Определение структуры**:
   - Определите структуру (назовем её `ArrayData`) со следующими полями:
     - `int n`: Размер массива.
     - `double* data`: Указатель на динамически выделенный массив с исходными данными.
     - `double* sortedData`: Указатель на динамически выделенный массив с отсортированными данными.
     - `double average`: Среднее значение элементов массива.

    - `double standardDeviation`: Стандартное отклонение элементов массива.
2. **Чтение из файла и инициализация структуры**:
    - Программа должна запросить у пользователя имя входного файла.
    - Программа должна открыть и прочитать размер массива `n` (целое число) из *первой строки* файла.
    - На основе прочитанного `n`, выделите память для полей `data` и `sortedData` структуры `ArrayData`.
    - Программа должна читать `n` чисел с плавающей точкой типа `double` из последующих строк файла и сохранить их в поле `data` структуры.
    - Создайте копию массива `data` в массиве `sortedData`. Это необходимо для того чтобы сортировка не изменяла исходные данные.
    - Если при чтении файла возникают ошибки (например, файл не существует, файл содержит некорректные данные или недостаточно чисел), программа должна вывести сообщение об ошибке и завершиться.
3. **Вычисление статистики и сохранение в структуру**:
    - Реализуйте функцию `double calculateAverage(double arr[], int n)`, которая принимает массив `arr` и его размер `n` в качестве аргументов и возвращает среднее значение элементов массива. Используйте эту функцию для вычисления среднего значения массива, хранящегося в поле `data` структуры, и сохраните результат в поле `average` структуры.
    - Реализуйте функцию `double calculateStandardDeviation(double arr[], int n, double average)`, которая принимает массив `arr`, его размер `n` и *уже вычисленное* среднее значение `average` в качестве аргументов и возвращает стандартное отклонение элементов массива. Используйте эту функцию для вычисления стандартного отклонения массива, хранящегося в поле `data` структуры, используя поле `average` этой же структуры, и сохраните результат в поле `standardDeviation` структуры.
    - Выведите на экран среднее значение и стандартное отклонение, полученные из полей соответствующей структуры. Вывод значений должен быть с двумя знаками после запятой.
4. **Сортировка и запись в файл**:
    - Реализуйте функцию `void sortArray(double arr[], int n)`, которая сортирует массив `arr` по *возрастанию*. Можно использовать любой алгоритм сортировки (например, пузырьковую сортировку, сортировку вставками, сортировку выбором). Отсортируйте массив `sortedData`, хранящийся в структуре.
    - Программа должна запросить у пользователя имя выходного файла.
    - Запишите отсортированный массив (из поля `sortedData` структуры) в указанный файл. Каждое число должно быть записано на отдельной строке.
5. **Освобождение памяти**:
    - Перед завершением программы, освободите память, выделенную для полей `data` и `sortedData` структуры.

Формулы:

- **Среднее значение (average):** `average = (sum of all elements) / (number of elements)`
- **Стандартное отклонение (standard deviation):** `standard deviation = sqrt( (sum of (x_i - average)^2) / (number of elements) )`

Указания:

- Используйте `iostream` и `fstream` для работы с файлами (в C++) или `stdio.h` для работы с файлами (в C).
- Используйте тип данных `double` для массива и всех вычислений.
- Используйте библиотеку `<cmath>` (C++) или `<math.h>` (C) для функции `sqrt()` и `pow()`.
- Функции `calculateAverage` и `calculateStandardDeviation` должны быть реализованы так же, как и в предыдущих заданиях.
- Обязательно проверяйте успешность открытия и чтения файлов.
- Обязательно освобождайте выделенную память после использования.
- Придерживайтесь хорошего стиля кодирования.

Пример содержимого входного файла (`data.txt`):

```
5
1.0
2.5
3.7
4.2
5.9
```

Ответ:

```
#include <iostream>
#include <fstream>
#include <cmath>
#include <iomanip>
#include <string>
#include <algorithm>

// Определение структуры для хранения данных массива
struct ArrayData {
    int n;
    double* data;
    double* sortedData;
    double average;
    double standardDeviation;
};

// Функция для вычисления среднего значения
double calculateAverage(double arr[], int n) {
    double sum = 0.0;
    for (int i = 0; i < n; ++i) {
        sum += arr[i];
    }
    return sum / n;
}

// Функция для вычисления стандартного отклонения
double calculateStandardDeviation(double arr[], int n, double average) {
    double sumOfSquaredDifferences = 0.0;
    for (int i = 0; i < n; ++i) {
        sumOfSquaredDifferences += pow(arr[i] - average, 2);
    }
    return sqrt(sumOfSquaredDifferences / n);
}

// Функция для сортировки массива (используем std::sort)
void sortArray(double arr[], int n) {
    std::sort(arr, arr + n);  // Используем стандартную функцию сортировки
}

int main() {
    std::string inputFileName, outputFileName;
    ArrayData arrayData;
    arrayData.data = nullptr;
    arrayData.sortedData = nullptr;

    // Шаг 1: Получение имени входного файла
```

```
    std::cout << "Введите имя входного файла: ";
    std::cin >> inputFileName;

    // Открываем входной файл
    std::ifstream inputFile(inputFileName);
    if (!inputFile.is_open()) {
        std::cerr << "Ошибка: Не удалось открыть файл " << inputFileName << std::endl;
        return 1;
    }

    // Читаем размер массива из файла
    if (!(inputFile >> arrayData.n)) {
        std::cerr << "Ошибка: Не удалось прочитать размер массива из файла." << std::endl;
        inputFile.close();
        return 1;
    }

    if (arrayData.n <= 0) {
       std::cerr << "Ошибка: Некорректный размер массива (N <= 0)." << std::endl;
       inputFile.close();
       return 1;
    }


    // Выделяем память для массивов data и sortedData
    arrayData.data = new double[arrayData.n];
    arrayData.sortedData = new double[arrayData.n];

    // Читаем элементы массива из файла
    for (int i = 0; i < arrayData.n; ++i) {
        if (!(inputFile >> arrayData.data[i])) {
            std::cerr << "Ошибка: Не удалось прочитать элемент " << i + 1 << " из файла." <<
std::endl;
            inputFile.close();
            delete[] arrayData.data;
            delete[] arrayData.sortedData;
            return 1;
        }
        arrayData.sortedData[i] = arrayData.data[i]; // Копируем данные в sortedData
    }

    // Закрываем входной файл
    inputFile.close();

    // Шаг 2: Вычисление среднего значения и стандартного отклонения
    arrayData.average = calculateAverage(arrayData.data, arrayData.n);
    arrayData.standardDeviation = calculateStandardDeviation(arrayData.data, arrayData.n,
arrayData.average);

    std::cout << std::fixed << std::setprecision(2);
    std::cout << "Среднее значение: " << arrayData.average << std::endl;
    std::cout << "Стандартное отклонение: " << arrayData.standardDeviation << std::endl;
```

```
    // Шаг 3: Получение имени выходного файла
    std::cout << "Введите имя выходного файла: ";
    std::cin >> outputFileName;

    // Сортируем массив sortedData
    sortArray(arrayData.sortedData, arrayData.n);

    // Открываем выходной файл
    std::ofstream outputFile(outputFileName);
    if (!outputFile.is_open()) {
        std::cerr << "Ошибка: Не удалось открыть файл " << outputFileName << " для записи."
<< std::endl;
        delete[] arrayData.data;
        delete[] arrayData.sortedData;
        return 1;
    }

    // Записываем отсортированный массив в файл
    for (int i = 0; i < arrayData.n; ++i) {
        outputFile << arrayData.sortedData[i] << std::endl;
    }

    // Закрываем выходной файл
    outputFile.close();

    std::cout << "Отсортированный массив записан в файл " << outputFileName << std::endl;

    // Освобождаем память
    delete[] arrayData.data;
    delete[] arrayData.sortedData;

    return 0;
}
```

Задание по теме 11:

Цель: Освоить базовые инструменты и функции Inkscape для создания и редактирования векторной графики.

Описание задания:

1. **Создание нового документа:**
   - Откройте Inkscape и создайте новый документ. Выберите подходящий формат (например, A4).
   - Установите единицы измерения на миллиметры (мм).
2. **Создание базовых фигур:**
   - Используя инструмент "Прямоугольник и квадрат", нарисуйте квадрат размером 50x50 мм. Установите цвет заливки и обводки по вашему выбору.
   - Используя инструмент "Эллипс и круг", нарисуйте круг диаметром 40 мм. Установите другой цвет заливки и обводки.
   - Используя инструмент "Звезда и многоугольник", создайте пятиконечную звезду. Измените количество углов, радиус и другие параметры по своему желанию. Установите свой цвет заливки и обводки.
3. **Редактирование фигур:**
   - Используя инструмент "Выделение и трансформация объектов", переместите фигуры так, чтобы они частично перекрывали друг друга.

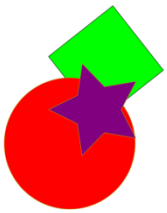
- Измените размер и угол поворота каждой фигуры. Попробуйте изменять размеры, удерживая клавишу Ctrl (для пропорционального изменения) и Shift (для масштабирования относительно центра).
- Измените порядок расположения объектов (передний план/задний план) с помощью контекстного меню ("Поднять", "Опустить").
- Используя инструмент "Редактирование узлов", измените форму одной из фигур (например, измените квадрат, чтобы получился ромб или трапеция). Попробуйте добавить или удалить узлы.

4. **Работа с контурами и заливками:**
   - Измените цвет заливки и обводки для каждой фигуры.
   - Установите толщину обводки (например, 2 мм).
   - Измените тип обводки (например, пунктирная линия).
   - Используйте градиентную заливку для одной из фигур. Поэкспериментируйте с различными типами градиентов (линейный, радиальный) и цветами.
   - Используйте инструмент "Пипетка", чтобы скопировать цвет с одной фигуры на другую.
5. **Работа с текстом:**
   - Используя инструмент "Текст", добавьте текст на холст. Напишите свое имя и фамилию.
   - Измените шрифт, размер шрифта и цвет текста.
   - Добавьте эффект "тень" к тексту.
6. **Использование слоев:**
   - Создайте новый слой (меню "Слой" -> "Добавить слой").
   - Переместите текст на новый слой.
   - Заблокируйте слой с текстом, чтобы случайно не изменить его.
   - Сделайте слой с текстом невидимым.
7. **Группировка объектов:**
   - Выделите все фигуры и сгруппируйте их (меню "Объект" -> "Сгруппировать").
   - Переместите сгруппированный объект.
   - Разгруппируйте объект (меню "Объект" -> "Разгруппировать").
8. **Сохранение файла:**
   - Сохраните работу в двух форматах:
     - **Формат Inkscape SVG (.svg):** Это основной формат Inkscape, позволяющий сохранить все элементы векторной графики для дальнейшего редактирования.
     - **Формат Portable Document Format (.pdf):** Это формат, который подходит для печати и обмена графикой, т.к. он сохраняет внешний вид документа независимо от устройства.

Критерии оценки:

- Правильное использование базовых инструментов Inkscape (прямоугольник, эллипс, звезда, текст).
- Умение редактировать фигуры (перемещение, масштабирование, поворот, изменение формы узлов).
- Умение работать с цветом, заливками и обводками.
- Использование слоев для организации работы.
- Умение группировать объекты.
- Правильное сохранение файла в требуемых форматах.
- Креативность и эстетичность работы.

Ответ:



итоговый рисунок

Критерий оценивания самодеятельной работы – результаты по каждой работе оформляться по указанным требованиям (смотрите в описании задания) и загружаться на образовательный портал ИГУ (https://educa.isu.ru/). Преподаватель оценивает задания, если все решено верно, студен получает зачет по задания, если имеются недочеты или ошибки, задание отправляться на доработку с указанием допущенных ошибок. Отчёт по переработанному заданию загружается на образовательный портал для повторного оценивания.

3.Оценочные средства для промежуточной аттестаци

Промежуточная аттестация проходит в форме экзамена (1 семестр), к которому допускаются студенты, выполнившие в полном объеме аудиторную нагрузку, самостоятельную работу. Студенты, имеющие задолженность, должны выполнить все обязательные виды деятельности.

Фонд оценочных средств для промежуточной аттестации включает:

- тестовые задания для экзамена.

Назначение оценочных средств: выявить сформированность компетенций ОПК-6, ОПК-7 (см. п. III).

Тестовое задание включает два варианта по 20 вопросов по всем темам курса. К тесту допускаются студенты, сдавшие все домашние задания и получившие по каждому заданию зачет.

Критерий оценивания тестового экзаменационного задания

Критерий оценки за тестирование

№	Тип задания	Критерии оценки	Результат оценивания
1	Задание закрытого типа на установление соответствия	Считается верным, если правильно установлены все соответствия (позиции одного столбца верно соотнесены с позициями другого столбца)	Полное совпадение с верным ответом – 1 балл Все остальные случаи – 0 баллов
2	Задание закрытого типа на установление последовательности	Считается верным, если правильно указана вся последовательность цифр	Полное совпадение с верным ответом – 1 балл Все остальные случаи – 0 баллов
3	Задание комбинированного типа с выбором одного верного ответа из четырех предложенных и обоснованием выбора	Считается верным, если правильно указана цифра (буква) правильного ответа и приведены корректные аргументы, используемые при выборе ответа	Полное совпадение с верным ответом – 1 балл Все остальные случаи – 0 баллов
4	Задание комбинированного типа с выбором нескольких верных ответов из четырех предложенных и обоснованием выбора	Считается верным, если правильно указаны цифры (буквы) правильного ответа и приведены корректные аргументы, используемые при выборе ответа	Полное совпадение с верным ответом – 1 балл Все остальные случаи – 0 баллов

5	Задание открытого типа с развернутым ответом	Считается верным, если ответ совпадает с эталонным ответом по содержанию и полноте	Полное соответствие эталонному ответу – 1 балл Все остальные случаи – 0 баллов

Критерий оценки за тестирование

Система получения баллов за тестирование

Оценка	**критерий**
отлично	18 и более баллов
хорошо	16 – 17 баллов
удовлетворительно	15 – 13 баллов
неудовлетворительно	12 баллов и менее

3.1 Оценочные материалы для промежуточной аттестации (экзамена)

Тестирование (Вариант 1).

Индекс и содержание формируемой компетенции	Индикаторы компетенций	Тестовые задания для промежуточной аттестации
ОПК-6 Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения	*ИДК ОПК-6.1* Знает принципы создания компьютерных программ, используемых в биоинформатике и биоинженерии *ИДК ОПК-6.2* Использует современные IT-технологии при сборе, анализе, обработке и представлении информации *ИДК ОПК-6.3* Использует навыки создания компьютерных программ, баз данных и иные программных продуктов, используемых в биоинженерии и биоинформатике	**Задание комбинированного типа с выбором одного или нескольких верных ответов из четырех предложенных и аргументацией выбора** ***Вопрос 1.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* Какие из перечисленных типов данных являются целочисленными в C/C++? a) float b) int c) double d) char Варианты ответов: 1. Только b 2. b и d 3. a и c 4. a, b, c и d Правильный ответ: 2 (b и d) Аргументация: • int - это основной целочисленный тип, используемый для хранения целых чисел. • char - хоть и используется для хранения символов, в C/C++ он также является целочисленным типом, так как символы представлены числовыми кодами (обычно ASCII). • float и double - типы данных с плавающей точкой, предназначенные для хранения чисел с десятичной частью. ***Вопрос 2.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* Какие из следующих операторов используются для логического "И" в C/C++? a) & b) && c) | d) || Варианты ответов: 1. Только a 2. Только b 3. a и c 4. b и d Правильный ответ: 2 (Только b) Аргументация:
ОПК-7 Способен понимать принципы работы современных	*ИДК ОПК-7.1* Демонстрирует теоретические и практические	

информационных технологий и использовать их для решения задач профессиональной деятельности	навыки использования современных информационных технологий в области профессианальной деятельности *ИДК ОПК-7.1* Использует современные информационные технологии в рамках освоения материала и реализации задач в области профессиональной деятельности	• && - это оператор логического "И". Он возвращает true только если оба операнда имеют значение true. • & - это побитовый оператор "И". Он выполняет побитовую операцию "И" между операндами. • || - это оператор логического "ИЛИ". • | - это побитовый оператор "ИЛИ". ***Вопрос 3.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* Что из следующего верно относительно указателей в C/C++? a) Указатель хранит адрес переменной. b) Указатель может хранить только адреса целых чисел. c) Указатель должен быть инициализирован при объявлении. d) Можно выполнять арифметические операции с указателями. Варианты ответов: 1. Только a 2. a и d 3. b и c 4. a, b и d Правильный ответ: 2 (a и d) Аргументация: • a - Указатель хранит адрес в памяти, где расположена переменная. • d - Можно выполнять арифметические операции (сложение, вычитание) с указателями для перемещения по памяти. Например, для доступа к элементам массива. • b - Указатель может хранить адрес переменной любого типа данных. • c - Инициализация указателя при объявлении не обязательна, но рекомендуется для избежания ошибок (обычно инициализируют nullptr или NULL). ***Вопрос 4.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* Какие из следующих утверждений верны относительно функций в C/C++? a) Функция может возвращать несколько значений. b) Функция может быть рекурсивной. c) Функция должна обязательно принимать аргументы. d) Функция может быть объявлена внутри другой функции. Варианты ответов: 1. Только b 2. b и d 3. a и c 4. a, b и d Правильный ответ: 1 (Только b) Аргументация: • b - Функция может вызывать саму себя, это называется рекурсией. • a - Функция в C/C++ может возвращать только одно значение (или void, если не возвращает ничего). Для возврата

		нескольких значений используются структуры, классы, массивы или указатели. • c - Функция может не принимать аргументы (аргументы могут отсутствовать). • d - В C/C++ (до C++11) функции обычно не объявляются внутри других функций, за исключением лямбда-функций (в C++11 и новее). ***Вопрос 5.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* Какие из перечисленных типов данных гарантированно имеют наименьший размер в памяти (среди перечисленных) в стандарте C? a) int b) short c) long d) char Варианты ответов: 1. Только a 2. Только d 3. b и d 4. a, b, c и d Правильный ответ: 2 (Только d) Аргументация: • Стандарт C гарантирует, что char занимает наименьшее количество памяти, достаточное для хранения одного символа. • Размер int, short и long зависит от конкретной архитектуры и компилятора, но char всегда будет наименьшим. **Вопрос 6.** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* Что из следующего не верно относительно указателей в C? a) Указатель хранит адрес переменной. b) void* можно использовать для указания на любой тип данных. c) Указатель всегда занимает 4 байта памяти. d) Можно использовать указатели для динамического выделения памяти. Варианты ответов: 1. Только a 2. Только c 3. b и d 4. a и d Правильный ответ: 2 (Только c) Аргументация: • a - Указатель хранит адрес в памяти, где расположена переменная. Это фундаментальное свойство указателей. • b - void* - это универсальный указатель, который может указывать на данные любого типа. Требуется приведение типа при использовании void*. • c - Размер указателя зависит от архитектуры процессора (32-битная или 64-битная). Он может быть 4 байта (на 32-битных системах) или 8 байт (на 64-битных системах). Это утверждение *не* всегда верно.

		• d – Оператор new используются для динамического выделения памяти в C, и они возвращают указатели на выделенную область. ***Вопрос 7.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* Какие из следующих утверждений верны относительно функций в C? a) Функция может возвращать несколько значений через структуру. b) Функция может быть рекурсивной. c) Функция должна обязательно принимать аргументы. d) В C функции могут быть перегружены (иметь одинаковое имя, но разные аргументы). Варианты ответов: 1. Только b 2. a и b 3. a и c 4. a, b и d Правильный ответ: 2 (a и b) Аргументация: • a - Функция в C может вернуть структуру, которая содержит несколько полей, тем самым косвенно возвращая несколько значений. • b - Функция может вызывать саму себя, это называется рекурсией. • c - Функция может не принимать аргументы (аргументы могут отсутствовать). • d - Перегрузка функций *не* поддерживается в C. Это особенность C++. ***Вопрос 8.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* Какой способ выделения памяти позволяет создать массив, размер которого определяется во время выполнения программы? • A) Объявление массива с переменной в качестве размера: int arr[n]; где n - переменная. • B) Использование оператор new для выделения динамической памяти • C) Использование ключевого слова dynamic. • D) Объявление массива как глобальной переменной. Правильные ответы: B Аргументация: • A) Объявление массива с переменной в качестве размера: int arr[n]; где n - переменная. – это особенность, появившаяся в стандарте C99 и называется Variable Length Array (VLA). Не все компиляторы поддерживают VLA, и их использование имеет свои ограничения. • B) Использование оператор new. - Это *правильный* и наиболее распространенный способ динамического выделения памяти для массивов (и вообще, любых структур данных) в C и C++. Размер блока может быть определен во время выполнения.

		• C) Использование ключевого слова dynamic. - В C нет ключевого слова dynamic для выделения памяти. Для динамического выделения памяти используется оператор new.
		• D) Объявление массива как глобальной переменной. - Объявление массива как глобальной переменной не позволяет определить размер массива во время выполнения. Размер глобального массива должен быть известен во время компиляции.
		
		Вопрос 9.
		Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.
		Что произойдет, если при использовании функции scanf для ввода данных в массив, пользователь введет больше элементов, чем размер массива? Например, массив int arr[5] и пользователь вводит 7 чисел.
		• A) scanf автоматически увеличит размер массива.
		• B) scanf запишет только первые 5 элементов, а остальные будут проигнорированы.
		• C) scanf запишет данные за пределы выделенной памяти для массива, что может привести к непредсказуемым последствиям.
		• D) Программа выдаст ошибку компиляции.
		Правильные ответы: C
		Аргументация:
		• A) scanf автоматически увеличит размер массива. - C не предоставляет автоматического расширения массивов.
		• B) scanf запишет только первые 5 элементов, а остальные будут проигнорированы. - К сожалению, scanf *не* имеет встроенной защиты от переполнения буфера. Он будет продолжать записывать данные в память, даже если массив переполнен.
		• C) scanf запишет данные за пределы выделенной памяти для массива, что может привести к непредсказуемым последствиям. - Это *именно то*, что произойдет. scanf запишет данные за пределы массива, что может привести к повреждению данных, ошибкам сегментации, или другому непредсказуемому поведению программы. Это серьезная проблема безопасности, известная как "переполнение буфера".
		• D) Программа выдаст ошибку компиляции. - Компилятор не может предвидеть, сколько данных введет пользователь во время выполнения, поэтому ошибка компиляции не произойдет.
		
		Вопрос 10.
		Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.
		Какое значение будет содержать arr[2] после выполнения следующего кода?
		int arr[5] = {0};
		arr[2] = arr[0] + 5;
		arr[0] = 10;
		• A) 0
		• B) 5
		• C) 10

		• D) 15 Правильные ответы: B Аргументация: • A) 0 - Неверно. Изначально arr[2] имеет значение 0, но потом его значение меняется. • B) 5 - Верно. Изначально все элементы массива инициализированы нулями. Затем arr[2] присваивается значение arr[0] + 5, что равно 0 + 5 = 5. Изменение значения arr[0] после этого не влияет на значение arr[2]. • C) 10 - Неверно. arr[0] присваивается значение 10, но это не влияет на arr[2] после того, как arr[2] было присвоено значение. • D) 15 - Неверно. Это было бы верно, если бы arr[2] присваивалось значение *после* изменения arr[0]. ***Вопрос 11.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* В каждом вопросе выберите один или несколько правильных ответов из предложенных вариантов. Обоснуйте свой выбор кратким объяснением. 1. Какой из следующих операторов является оператором цикла с предусловием в C/C++? • A) for • B) while • C) do...while • D) switch Ваш ответ: B) while Обоснование: Цикл while проверяет условие *перед* выполнением блока кода. Если условие ложно с самого начала, блок кода не выполнится ни разу. for цикл может иметь предусловие, но он также включает инициализацию и инкремент/декремент. do...while - это цикл с постусловием. switch - это оператор выбора, а не цикл. ***Вопрос 12.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* Что произойдет, если условие в цикле while всегда истинно? • A) Программа выдаст ошибку компиляции. • B) Программа не запустится. • C) Произойдет бесконечный цикл. • D) Цикл выполнится ровно один раз. Правельный ответ: C) Произойдет бесконечный цикл. Обоснование: Если условие цикла while всегда истинно (например, while(1) или while(true)), то цикл будет выполняться бесконечно, пока не будет прерван вручную или пока не произойдет ошибка, например, переполнение памяти. ***Вопрос 13.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.*

		В каком из следующих случаев наиболее уместно использовать цикл for? • A) Когда количество итераций цикла заранее неизвестно. • B) Когда тело цикла должно выполниться хотя бы один раз. • C) Когда необходимо выполнить блок кода бесконечное количество раз. • D) Когда количество итераций цикла известно заранее или может быть легко вычислено. Правильный ответ: D) Когда количество итераций цикла известно заранее или может быть легко вычислено. Обоснование: Цикл for обычно используется, когда известно количество итераций, так как он включает инициализацию, условие и инкремент/декремент в одной строке, что делает его удобным для таких случаев. while лучше подходит, когда количество итераций заранее неизвестно, а do...while - когда нужно гарантировать хотя бы однократное выполнение. **Задание закрытого типа на установление соответствия** ***Вопрос14.*** *Прочитайте вопрос и установите соответствие.* Установите соответствие между операторами условного перехода в C/C++ (Левая колонка) и их описаниями (Правая колонка). Каждому оператору соответствует только одно описание. Левая колонка (Операторы) 1. if 2. if...else 3. if...else if...else 4. switch Правая колонка (Описания) A. Позволяет выбрать один из нескольких блоков кода для выполнения в зависимости от значения целочисленного выражения. B. Выполняет блок кода только в том случае, если указанное условие истинно. C. Предоставляет альтернативный блок кода для выполнения, если указанное в if условие ложно. D. Предоставляет несколько взаимоисключающих условий, позволяя выбрать один из нескольких блоков кода для выполнения. Последний else блок выполняется, если ни одно из предыдущих условий не было истинным. Установите соответствие: • 1 - ? • 2 - ? • 3 - ? • 4 - ? Правильный ответ: • 1 - B • 2 - C • 3 - D

- 4 - A

Вопрос 15.

Прочитайте вопрос и установите соответствие.

Установите соответствие между примерами кода или описаниями (Левая колонка) и соответствующими особенностями использования операторов цикла в C/C++ (Правая колонка). Каждому элементу левой колонки соответствует только один элемент правой колонки.

Левая колонка (Примеры кода/Описания)

1. Использование break для немедленного выхода из цикла.
2. Использование continue для пропуска текущей итерации и перехода к следующей.
3. Бесконечный цикл, который должен быть прерван с помощью break при выполнении определенного условия.
4. Эквивалент for (int i = 0; i < 10; ++i) { ... }, реализованный с использованием цикла while.

Правая колонка (Особенности/Примеры кода)

A. c++ int i = 0; while (i < 10) { // ... ++i; }

B. c++ while (true) { // ... if (condition) { break; } }

C. Цикл с ранним выходом (early exit)

D. Пропуск итерации цикла

Установите соответствие:

- 1 - ?
- 2 - ?
- 3 - ?
- 4 - ?

Правильный ответ:

- 1 - C
- 2 - D
- 3 - B
- 4 - A

Вопрос 16.

Прочитайте вопрос и установите соответствие.

Поставьте в соответствие математическую функцию библиотеки math.h из левого столбца с её наиболее точным описанием из правого столбца.

Функция	Описание
1. round(x)	A. Возвращает ближайшее целое число к x. Если x находится ровно посередине между двумя целыми числами, возвращает четное.

2. fmax(x, y)	B. Возвращает модуль разности между значениями x и y.
3. tan(x)	C. Возвращает наибольшее из двух значений x и y.
4. trunc(x)	D. Возвращает тангенс угла x (в радианах).
5. fdim(x, y)	E. Возвращает целую часть x, отбрасывая дробную часть.
6. acosh(x)	F. Возвращает обратный гиперболический косинус числа x.

Установите соответствие:

- 1 - ?
- 2 - ?
- 3 - ?
- 4 - ?
- 5 - ?
- 6 - ?

Правильный ответ:

- 1 - A
- 2 - C
- 3 - D
- 4 - E
- 5 - B
- 6 - F

Вопрос 17.

Прочитайте вопрос и установите соответствие.

Поставьте в соответствие типы данных C/C++ из левого столбца с их описанием из правого столбца.

Тип данных	Описание
1. int	A. Тип данных, представляющий собой строку символов.
2. float	B. Тип данных, представляющий логическое значение (истина или ложь).
3. char	C. Тип данных, представляющий целое число.
4. bool	D. Тип данных, представляющий число с плавающей точкой одинарной точности.

5. double	E. Тип данных, представляющий один символ.
6. std::string	F. Тип данных, представляющий число с плавающей точкой двойной точности.

Установите соответствие:
- 1 - ?
- 2 - ?
- 3 - ?
- 4 - ?
- 5 - ?
- 6 - ?

Правильный ответ:
- 1 - C
- 2 - D
- 3 - E
- 4 - B
- 5 - F
- 6 - A

Задание закрытого типа на установление последовательности

Вопрос 18.
Прочитайте вопрос и установите последовательность.
Расположите следующие действия в правильной последовательности, описывающей основной процесс использования функции в C/C++.
Действия:
A. Вызов функции: Использование имени функции с передачей аргументов (если необходимо) для выполнения ее кода.
B. Определение функции: Предоставление кода, который будет выполнен при вызове функции.
C. Объявление функции: Указание имени функции, возвращаемого типа и типов аргументов (если есть).
Установите последовательность:
? -> ? -> ?
Правильный ответ:
C -> B -> A

Вопрос 19.
Прочитайте вопрос и установите последовательность.
Расположите следующие действия в правильной последовательности при работе со структурой в C/C++, которая хранит

		числовое и текстовое значение. Действия: A. Присвоение значений членам структуры: Заполнение числового и текстового полей конкретного экземпляра структуры. B. Объявление структуры: Определение структуры с объявлением члена для хранения числового значения (например, int) и члена для хранения текстового значения (например, std::string или char[]). C. Создание экземпляра структуры: Выделение памяти для хранения одного объекта (экземпляра) определенной структуры. D. Использование значений членов структуры: Обращение к числовому и текстовому полям для чтения, вывода или других операций. Установите последовательность (например: A -> B -> C -> D): ? -> ? -> ? -> ? Правильный ответ: B -> C -> A -> D ***Вопрос 20.*** *Прочитайте вопрос и установите последовательность.* Ситуация: Необходимо прочитать из текстового файла "student.txt" информацию о студенте (имя, фамилию, возраст), записанную ранее, и вывести её на экран. Расположите следующие действия в правильной последовательности: 1. file.close(); 2. std::cout << "Имя: " << firstName << std::endl << "Фамилия: " << lastName << std::endl << "Возраст: " << age << std::endl; 3. std::string firstName, lastName; int age; 4. file >> firstName >> lastName >> age; 5. #include <iostream> 6. std::ifstream file("student.txt"); Установите последовательность (например: 1 -> 2 -> 3 -> 4 -> 5-> 6): ? -> ? -> ? -> ? -> ? -> ? -> ? Правильный ответ: 5 -> 6 -> 3 -> 4 -> 2 -> 1 ***Вопрос 21.*** *Прочитайте вопрос и установите последовательность.* Ситуация: Необходимо написать функцию, которая принимает на вход размер массива и возвращает указатель на динамически созданный и инициализированный массив целых чисел, где каждый элемент равен своему индексу. После использования необходимо освободить выделенную память в вызывающей функции.

Расположите следующие действия в правильной последовательности для *определения* функции createArray:

1. int* createArray(int size) {
2. for (int i = 0; i < size; ++i) { arr[i] = i; }
3. int* arr = new int[size];
4. return arr;
5. }

Установите последовательность (например: 1 -> 2 -> 3 -> 4 -> 5-> 6):

? -> ? -> ? -> ? -> ? -> ? -> ? -> ?

Правильный ответ:

1 -> 3 -> 2 -> 4 -> 5

Задание открытого типа с развернутым ответом

Вопрос 22.

Прочитайте вопрос и запишите развернутый обоснованный ответ.

Опишите синтаксис создания динамического целочисленного массива в C++?

Правильный ответ:

```
int size;
// Выделение памяти для массива из 'size' элементов типа int
int* myArray = new int[size];
// Освобождение выделенной памяти (ОБЯЗАТЕЛЬНО!)
delete[] myArray;
```

Вопрос 23.

Прочитайте вопрос и запишите развернутый обоснованный ответ.

Напишите код на языке C/C++, который принимает на вход целочисленный массив arr из 20 элементов и сортирует его по возрастанию. Используйте алгоритм сортировки пузырьком.

Правильный ответ:

```
// Сортировка пузырьком
int temp;
  for (int i = 0; i < 19; ++i) {
    for (int j = 0; j < 19 - i; ++j) {
      if (arr[j] > arr[j + 1]) {
        // Обмен элементов
        temp = arr[j];
        arr[j] = arr[j + 1];
        arr[j + 1] = temp;
      }
```

		}
		}
		
		Вопрос 24.
		Прочитайте вопрос и запишите развернутый обоснованный ответ.
		Перечислите основные встроенные (базовые) типы данных, для хранения числовых значений, доступные в языках программирования C и C++. Для каждого типа укажите его название, краткое описание.
		Правильный ответ:
		В языках C и C++ есть следующие основные встроенные типы данных:
		• **int**: Целочисленный тип. Используется для представления целых чисел (как положительных, так и отрицательных) без дробной части.
		• **float**: Вещественный тип с одинарной точностью. Используется для представления чисел с плавающей точкой (то есть чисел с дробной частью).
		• **double**: Вещественный тип с двойной точностью. Используется для представления чисел с плавающей точкой с большей точностью, чем float.
		
		Вопрос 25.
		Прочитайте вопрос и запишите развернутый обоснованный ответ.
		Опишите синтаксис цикла for в языке программирования C/C++?
		Правильный ответ:
		Синтаксис цикла for в C/C++ выглядит следующим образом:
		for (инициализация; условие; итерация) {
		// Код, который будет выполняться в каждой итерации цикла
		}

Тестирование (Вариант 2).

Индекс и содержа ние формируемой компетенции	**Индикаторы компетенций**	**Тестовые задания для промежуточной аттестации**
ОПК-6 Способен разрабатывать алгоритмы и компьютерные программы, пригодные для	*ИДК ОПК-6.1* Знает принципы создания компьютерных программ, используемых в биоинформатике и	**Задание комбинированного типа с выбором одного или нескольких верных ответов из четырех предложенных и аргументацией выбора** ***Вопрос 1.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* Что из перечисленного является корректным способом объявления целочисленной переменной в C?

практического применения	биоинженерии	• A) int x; • B) integer x; • C) x : integer; • D) define x int; Правильные ответы: A Аргументация: • A) int x; - Это стандартный и правильный способ объявления целочисленной переменной в C. Сначала указывается тип данных (int), затем имя переменной (x), и в конце ставится точка с запятой. • B) integer x; - В C нет типа данных integer. Правильный тип для целых чисел - int. • C) x : integer; - Это синтаксис, используемый в некоторых других языках программирования (например, Pascal), но не в C. • D) define x int; - Это препроцессорная директива, используемая для определения макросов. Хотя это можно использовать для создания псевдонима int, это не стандартный способ объявления переменной и может привести к путанице. Кроме того, даже если бы это работало, переменная не была бы объявлена *фактически* до ее использования. ***Вопрос 2.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* Что из перечисленного верно относительно указателей в C? • A) Указатель хранит адрес переменной в памяти. • B) Указатель может указывать только на переменные типа int. • C) Указатель объявляется с помощью символа *. • D) NULL - это специальное значение, которое может быть присвоено указателю, чтобы обозначить, что он не указывает ни на какой объект. Правильные ответы: A, C, D Аргументация: • A) Указатель хранит адрес переменной в памяти. - Это фундаментальное определение указателя. Он "указывает" на место в памяти, где хранится значение переменной. • B) Указатель может указывать только на переменные типа int. - Это неверно. Указатели могут указывать на переменные любого типа данных (int, char, float, структуры, и т.д.). Важно, чтобы тип указателя соответствовал типу переменной, на которую он указывает. • C) Указатель объявляется с помощью символа *. - Действительно, символ * используется при объявлении указателя. Например, int *ptr; объявляет указатель ptr на целое число. • D) NULL - это специальное значение, которое может быть присвоено указателю, чтобы обозначить, что он не указывает ни на какой объект. - NULL (или 0) используется для обозначения "нулевого" указателя, то есть указателя, который в данный момент не указывает ни на какой допустимый адрес в памяти. Это полезно для избежания ошибок разыменования (попытки доступа к памяти по недействительному адресу).
	ИДК ОПК-6.2 Использует современные IT-технологии при сборе, анализе, обработке и представлении информации	
ОПК-7 Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности	*ИДК ОПК-6.3* Использует навыки создания компьютерных программ, баз данных и иные программных продуктов, используемых в биоинженерии и биоинформатике	
	ИДК ОПК-7.1 Демонстрирует теоретические и практические навыки использования современных информационных технологий в области профессианальной деятельности	
	ИДК ОПК-7.1 Использует современные информационные технологии в рамках освоения материала и реализации задач в области	

	профессиональной деятельности	***Вопрос 3.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* Какие из следующих функций являются частью стандартной библиотеки ввода/вывода C (stdio.h)? • A) printf • B) scanf • C) sqrt • D) fopen Правильные ответы: A, B, D Аргументация: • A) printf - Функция printf используется для форматированного вывода данных на стандартный вывод (обычно экран). Она является ключевой частью библиотеки stdio.h. • B) scanf - Функция scanf используется для форматированного ввода данных со стандартного ввода (обычно клавиатуры). Она также входит в stdio.h. • C) sqrt - Функция sqrt вычисляет квадратный корень числа. Она определена в библиотеке math.h, а не в stdio.h. • D) fopen - Функция fopen используется для открытия файла. Она является частью библиотеки stdio.h и позволяет работать с файлами для чтения и записи. ***Вопрос 4.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* Какие из следующих утверждений о массивах в C являются верными? • A) Индексы элементов массива начинаются с 1. • B) Все элементы массива должны быть одного типа данных. • C) Размер массива должен быть известен во время компиляции, если массив объявлен локально (внутри функции). • D) При передаче массива в функцию, передается копия массива. Правильные ответы: B, C Аргументация: • A) Индексы элементов массива начинаются с 1. - Неверно. В C (как и во многих других языках), индексы массивов начинаются с 0. Первый элемент массива имеет индекс 0, второй - 1, и так далее. • B) Все элементы массива должны быть одного типа данных. - Верно. Массив в C - это упорядоченный набор элементов *одного и того же* типа данных (например, int, float, char, или даже структуры). • C) Размер массива должен быть известен во время компиляции, если массив объявлен локально (внутри функции). - Верно, для массивов, объявленных локально, размер должен быть константой времени компиляции. Можно использовать динамическое выделение памяти (malloc), если размер массива заранее неизвестен. • D) При передаче массива в функцию, передается копия массива. - Неверно. При передаче массива в функцию, передается *указатель* на первый элемент массива. Это означает, что функция может изменять исходный массив. Копия массива *не* создается.

Вопрос 5.

Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.

Какой из следующих способов инициализации массива int arr[5] является корректным?

- A) int arr[5] = {1, 2, 3, 4, 5};
- B) int arr[5] = {1, 2, 3};
- C) int arr[5]; arr = {1, 2, 3, 4, 5};
- D) int arr[] = {1, 2, 3, 4, 5};

Правильные ответы: A, B, D

Аргументация:

- A) int arr[5] = {1, 2, 3, 4, 5}; - Это правильный способ инициализации массива при объявлении. Все 5 элементов инициализируются заданными значениями.
- B) int arr[5] = {1, 2, 3}; - Это тоже правильный способ. В этом случае, первые три элемента инициализируются значениями 1, 2 и 3, а остальные элементы (в данном случае, arr[3] и arr[4]) инициализируются нулями по умолчанию.
- C) int arr[5]; arr = {1, 2, 3, 4, 5}; - Это *не* корректно. Вы не можете присвоить массив таким образом после объявления. Оператор присваивания {1, 2, 3, 4, 5} можно использовать только во время объявления массива. В этом случае потребуется использовать цикл, чтобы присвоить значения каждому элементу по отдельности.
- D) int arr[] = {1, 2, 3, 4, 5}; - Это также правильно. Если вы инициализируете массив при объявлении, опуская размерность, компилятор определит размер массива на основе количества предоставленных инициализаторов.

Вопрос 6.

Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.

Что произойдет, если вы попытаетесь обратиться к элементу массива int arr[10] с индексом 10 (например, arr[10])?

- A) Программа выдаст ошибку компиляции.
- B) Программа выдаст ошибку времени при выполнении.
- C) Программа продолжит работу, но результат будет непредсказуемым.
- D) Программа автоматически увеличит размер массива.

Правильные ответы: B, C

Аргументация:

- A) Программа выдаст ошибку компиляции. - Компилятор обычно *не* обнаруживает выход за границы массива во время компиляции (особенно если индекс является переменной). Поэтому ошибки компиляции, скорее всего, не будет.
- B) Программа выдаст ошибку времени выполнения (segmentation fault или подобное). - Это *один из* возможных результатов. Обращение к памяти за пределами выделенной области может привести к ошибке сегментации (segmentation fault) или другому типу ошибки времени выполнения, которая завершит программу.
- C) Программа продолжит работу, но результат будет непредсказуемым. - Это *другой* возможный результат. В C нет автоматической проверки границ массива. Обращение к arr[10] может привести к чтению или записи в

		область памяти, которая не принадлежит массиву. Это может привести к непредсказуемым результатам, повреждению данных, зависанию программы или другим неожиданным проблемам. Это один из самых опасных типов ошибок в C, поскольку их трудно отлавливать. • D) Программа автоматически увеличит размер массива. - Это *абсолютно неверно*. C не предоставляет автоматического расширения массивов. Размер массива фиксируется при его объявлении. ***Вопрос 7.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* Какие из следующих утверждений верны при передаче массива в функцию в C? • A) Внутри функции можно узнать размер массива с помощью оператора sizeof. • B) Функция получает копию массива, поэтому изменения внутри функции не повлияют на исходный массив. • C) Функция получает указатель на первый элемент массива. • D) Размер массива необходимо передавать в функцию отдельным параметром (если требуется знать его размер). Правильные ответы: C, D Аргументация: • A) Внутри функции можно узнать размер массива с помощью оператора sizeof. - Это *неверно*. Внутри функции, когда массив передается как аргумент, он "преобразуется" в указатель на свой первый элемент. Оператор sizeof вернет размер указателя, а не размер исходного массива. • B) Функция получает копию массива, поэтому изменения внутри функции не повлияют на исходный массив. - Это *неверно*. Как уже говорилось, функция получает указатель на первый элемент массива. Следовательно, изменения, сделанные внутри функции (например, изменение значения элемента массива), *повлияют* на исходный массив. • C) Функция получает указатель на первый элемент массива. - Это *верно*. Это основной механизм передачи массивов в функции в C. • D) Размер массива необходимо передавать в функцию отдельным параметром (если требуется знать его размер). - Это *верно* и является хорошей практикой. Поскольку sizeof не работает корректно внутри функции (см. пункт A), если функции нужно знать размер массива, его нужно передать как отдельный аргумент. Это особенно важно, если функция должна обрабатывать все элементы массива. ***Вопрос 8.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* В каком из следующих циклов гарантируется хотя бы однократное выполнение тела цикла? • A) for • B) while • C) do...while • D) Ни в одном из перечисленных Ваш ответ: C) do...while Обоснование: Цикл do...while сначала выполняет блок кода, а затем проверяет условие. Таким образом, тело цикла

		всегда выполняется хотя бы один раз. Циклы for и while проверяют условие перед выполнением, поэтому тело цикла может не выполниться ни разу. ***Вопрос 9.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* Какой из следующих фрагментов кода выведет числа от 1 до 10 включительно? • A) for (int i = 0; i < 10; i++) { printf("%d ", i); } • B) for (int i = 1; i <= 10; i++) { printf("%d ", i); } • C) int i = 1; while (i <= 10) { printf("%d ", i); i++; } • D) int i = 1; do { printf("%d ", i); i++; } while (i < 10); Правельный ответ: B) for (int i = 1; i <= 10; i++) { printf("%d ", i); } и C) int i = 1; while (i <= 10) { printf("%d ", i); i++; } Обоснование: • A) Выведет числа от 0 до 9. Условие i < 10 останавливает цикл при i = 10, не включая его. • B) Выведет числа от 1 до 10, как и требуется. Условие i <= 10 позволяет включить 10. • C) Выведет числа от 1 до 10, как и требуется. Условие i <= 10 позволяет включить 10. • D) Выведет числа от 1 до 9. Условие i < 10 останавливает цикл при i = 10, не включая его, хотя тело цикла выполняется минимум один раз. ***Вопрос 10.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа*. Какие ключевые слова используются для прерывания цикла в C/C++? (Выберите все подходящие варианты) • A) break • B) continue • C) return • D) exit Правильный ответ: A) break и B) continue Обоснование: • break: Прерывает выполнение цикла и передает управление оператору, следующему за циклом. • continue: Прерывает текущую итерацию цикла и переходит к следующей итерации. • return: Выходит из текущей функции, в которой находится цикл. • exit: Завершает выполнение всей программы. ***Вопрос 11.*** *Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.* Какой оператор используется для вывода данных в консоль в C++? • A) scanf • B) printf • C) std::cout

- D) std::cin

Правельный ответ: C) std::cout
Обоснование: std::cout - это объект класса ostream, представляющий стандартный поток вывода (обычно консоль) в C++. std::cin используется для ввода. scanf и printf - это функции из C, которые также можно использовать в C++, но std::cout и std::cin являются частью стандартной библиотеки C++ и более типобезопасны.

Вопрос 12.
Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.
Какой объект класса используется для записи данных в файл в C++?

- A) std::cin
- B) std::cout
- C) std::ofstream
- D) std::ifstream

Ваш ответ: C) std::ofstream
Обоснование: std::ofstream (output file stream) - это класс для записи данных в файл. std::ifstream (input file stream) используется для чтения данных из файла. std::cin и std::cout предназначены для работы с консолью.

Вопрос13.
Прочитайте вопрос, выберите правильный вариант ответа и запишите аргументы, обосновывающие выбор ответа.
Какой заголовочный файл необходимо подключить для работы с файлами в C++?

- A) iostream
- B) fstream
- C) stdio.h
- D) iostream.h

Ваш ответ: B) fstream
Обоснование: Заголовочный файл <fstream> содержит определения классов ifstream, ofstream и fstream, которые используются для работы с файлами. iostream необходим для std::cin и std::cout. stdio.h - это заголовочный файл из C, предоставляющий функции для работы с файлами, но fstream предпочтительнее в C++. iostream.h - устаревший заголовочный файл.

Задание закрытого типа на установление соответствия

Вопрос 14.
Прочитайте вопрос и установите соответствие.
Установите соответствие между описанием логической задачи (левая колонка) и наиболее подходящим для её решения фрагментом кода на C/C++ (правая колонка). Каждый фрагмент кода может быть использован только один раз.

	Описание Задачи	**Варианты кода**
1	Определить, является ли число положительным,	A. if (x > 0) { printf("Положительное"); } else if (x < 0) {

	отрицательным или нулем, и вывести соответствующее сообщение.	printf("Отрицательное"); } else { printf("Ноль"); }
2	Проверить, находится ли введенное число в диапазоне от 10 до 20 (включительно). Если да, вывести "В диапазоне".	B. if (x >= 10 && x <= 20) { printf("В диапазоне"); } else {printf("Вне диапазона");}
3	Вывести "Четное", если число четное, и "Нечетное" в противном случае.	C. switch (x % 3) { case 0: printf("Делится на 3"); break; case 1: printf("Остаток 1"); break; default: printf("Остаток 2"); }
4	В зависимости от остатка от деления на 3 вывести разные сообщения.	D. if (x % 2 == 0) { printf("Четное") ; } else {printf("Нечетное");}

Установите соответствие:

- 1 - ?
- 2 - ?
- 3 - ?
- 4 - ?

Правильный ответ:

- 1 - A
- 2 - B
- 3 - D
- 4 - C

Вопрос 15.

Прочитайте вопрос и установите соответствие.

Установите соответствие между описаниями (Левая колонка) и соответствующими операторами цикла в C/C++ (Правая колонка). Каждому описанию соответствует только один оператор.

Левая колонка (Описания)

- Цикл, который гарантированно выполнится хотя бы один раз.
- Бесконечный цикл.
- Цикл, который позволяет выполнять блок кода до тех пор, пока заданное условие истинно.
- Цикл, который требует явного указания инициализации, условия и обновления счетчика (инкремента или декремента).

Правая колонка (Операторы)

A. for
B. while
C. do...while
D. for(;;)

Установите соответствие:

- 1 - ?
- 2 - ?
- 3 - ?
- 4 - ?

Правильный ответ:

- 1 - C
- 2 - D
- 3 - B
- 4 - A

Вопрос 16.

Прочитайте вопрос и установите соответствие.

Поставьте в соответствие математическую функцию библиотеки math.h с правильным примером её использования для достижения указанного результата из правого столбца.

Функция	Описание
1. log10(x)	A. double area = M_PI * r * r; (Вычислить площадь круга).
2. sinh(x)	B. double base10_log = log10(1000.0); (Вычислить десятичный логарифм числа 1000).
3. copysign(x, y)	C. double signed_value = copysign(magnitude, sign); (Перенести знак из переменной sign на переменную magnitude).
4. remainder(x,y)	D. double r = remainder(17.5, 5.0); (Вычислить остаток с плавающей точкой по стандарту IEEE 754).
5. modf(x, &iptr)	E. double fractional_part = modf(3.1415, &integer_part); (Разделить число с плавающей точкой на целую и дробную части).
6. Предопределенное значение M_PI	F. double hyp_sin = sinh(x); (Вычислить гиперболический синус числа x).

Установите соответствие:

- 1 - ?
- 2 - ?
- 3 - ?
- 4 - ?
- 5 - ?

- 6 - ?

Правильный ответ:

- 1 - B
- 2 - F
- 3 - C
- 4 - D
- 5 - E
- 6 - A

Вопрос 17.
Прочитайте вопрос и установите соответствие.
Поставьте в соответствие операторы C/C++ из левого столбца с действием, которое они выполняют, из правого столбца.

Оператор	Действие
1. ++	A. Присваивает значение правой стороны переменной слева.
2. ==	B. Возвращает остаток от деления.
3. =	C. Уменьшает значение переменной на 1.
4. %	D. Проверяет, равны ли два значения.
5. &&	E. Увеличивает значение переменной на 1.
6. --	F. Логическое "И": возвращает true, если оба операнда истинны.

Установите соответствие:

- 1 - ?
- 2 - ?
- 3 - ?
- 4 - ?
- 5 - ?
- 6 - ?

Правильный ответ:

- 1 - E
- 2 - D
- 3 - A

		• 4 - B • 5 - F • 6 - C Задание закрытого типа на установление последовательности *Вопрос 18.* *Прочитайте вопрос и установите последовательность.* Расположите следующие действия в правильной последовательности, описывающей процесс использования функции в C/C++. Действия: A. Вызов функции: Использование имени функции с передачей необходимых аргументов для выполнения её кода. B. Объявление функции (Function Declaration/Prototype): Указание имени функции, типа возвращаемого значения и типов аргументов. C. Определение функции (Function Definition): Предоставление фактического кода, который будет выполнен при вызове функции. Установите последовательность (например: A -> B -> C): ? -> ? -> ? -> ? Правильный ответ: B -> C -> A *Вопрос 19.* *Прочитайте вопрос и установите последовательность.* Расположите следующие действия в правильной последовательности, описывающей процесс использования структуры в C/C++. Действия: A. Доступ к члену структуры: Использование оператора . (точка) или -> (стрелка) для получения или изменения значения одного из полей (членов) структуры. B. Объявление структуры (Structure Declaration): Определение нового типа данных, состоящего из нескольких переменных (членов) различных типов. C. Создание экземпляра структуры (Structure Instantiation): Выделение памяти для переменной типа структуры. D. Инициализация членов структуры (Structure Initialization): Присвоение начальных значений членам структуры при создании экземпляра. Установите последовательность (например: A -> B -> C -> D): ? -> ? -> ? -> ? Правильный ответ: B -> C -> D -> A *Вопрос 20.*

		Прочитайте вопрос и установите последовательность.
		Задание: Необходимо записать имя, фамилию и возраст студента в текстовый файл "student.txt", каждый параметр с новой строки.
		Расположите следующие действия в правильной последовательности:
		• file << age << std::endl;
		• #include <iostream>
		• std::ofstream file("student.txt");
		• std::string firstName = "Иван"; std::string lastName = "Иванов"; int age = 20;
		• file.close();
		• file << firstName << std::endl; file << lastName << std::endl;
		Установите последовательность (например: 1 -> 2 -> 3 -> 4 -> 5-> 6):
		? -> ? -> ? -> ? -> ? -> ? -> ? -> ?
		Правильный ответ:
		2 -> 3 -> 4 -> 6 -> 1 -> 5
		
		Вопрос 21.
		Прочитайте вопрос и установите последовательность.
		Ситуация: Необходимо создать динамический массив целых чисел размера n, заполнить его квадратами чисел от 0 до n-1, распечатать значения элементов массива и затем освободить выделенную память.
		Расположите следующие действия в правильной последовательности:
		• for (int i = 0; i < n; ++i) { arr[i] = i * i; }
		• delete[] arr;
		• #include <iostream>
		• int* arr = new int[n];
		• for (int i = 0; i < n; ++i) { std::cout << arr[i] << " "; }
		• int n = 10; // Предположим, что размер массива равен 10
		Установите последовательность (например: 1 -> 2 -> 3 -> 4 -> 5-> 6):
		? -> ? -> ? -> ? -> ? -> ? -> ? -> ?
		Правильный ответ:
		3 -> 6 -> 4 -> 1 -> 5 -> 2
		
		Задание открытого типа с развернутым ответом
		Вопрос 22.
		Прочитайте вопрос и запишите развернутый обоснованный ответ.
		Напишите код на языке C/C++, который принимает на вход целочисленный массив из 20 элементов и возвращает максимальное значение, содержащееся в этом массиве.
		Правильный ответ:
		int maxVal = arr[0]; // Инициализируем значением элемента массива

```
  for (int i = 0; i < 20; ++i) {
    if (arr[i] > maxVal) {
      maxVal = arr[i];
    }
  }
```

Вопрос 23.

Прочитайте вопрос и запишите развернутый обоснованный ответ.

Напишите код на языке C/C++, который принимает на вход целочисленный массив из 20 элементов и сортирует его по убыванию методом пузырька.

Правильный ответ:

```
// Сортировка пузырьком (модифицированная для убывания)
int temp;
  for (int i = 0; i < 19; ++i) {
    for (int j = 0; j < 19 - i; ++j) {
      if (arr[j] < arr[j + 1]) {  // Изменено условие для убывания
        // Обмен элементов
        temp = arr[j];
        arr[j] = arr[j + 1];
        arr[j + 1] = temp;
      }
    }
  }
```

Вопрос 24.

Прочитайте вопрос и запишите развернутый обоснованный ответ.

Опишите синтаксис цикла while в языке программирования C/C++?

Правильный ответ:

Синтаксис цикла while в C/C++:

```
while (условие) {
  // Код, который будет выполняться, пока условие истинно
}
```

Вопрос 25.

Прочитайте вопрос и запишите развернутый обоснованный ответ.

		Напишите код на языке C/C++, который принимает на вход массив вещественных чисел из 20 элементов и возвращает минимальное значение, содержащееся в этом массиве.
		
		Правильный ответ:
		float minVal = arr[0]; // Инициализируем значением элемента массива
		for (int i = 0; i < 20; ++i) {
		if (arr[i] < minVal) {
		minVal = arr[i];
		}
		}

Разработчик:

_____ доцент Букин Ю.С.

(подпись)