**Комплект оценочных материалов по дисциплине
 «Программирование мехатронных и робототехнических систем»**

**Задания закрытого типа**

**Задания закрытого типа на выбор правильного ответа**

*Выберите один правильный ответ*

1. Каким образом должны быть объявлены переменные v и f, чтобы следующий код:

 f = cos;

 v = f(0);

корректно компилировался и выполнялся, в предположении что cos – стандартная функция из математической библиотеки <math.h>

А) int v, f;

Б) double v, f;

В) double (\*v)(double), f;

Г) double v, (\*f)(double);

Д) double (\*v)(double), (\*f)(double);

Правильный ответ: Г

Компетенции (индикаторы): ОПК-14.3

2. Каким образом в языке C определить матрицу с элементами типа double у которой M строк и N столбцов, чтобы к элементам можно было обращаться по двум индексам:

А) double matrix[M\*N];

Б) double matrix[M,N];

В) double matrix[M][N];

Г) double matrix[N][M];

Д) double matrix[N-1][M-1];

Правильный ответ: В

Компетенции (индикаторы): ОПК-14.3

3. Чему по стандарту эквивалентна с точки зрения конечного результата строка return(1); в теле функции main()?:

А) exit(0);

Б) abort();

В) exit(1);

Г) std::terminate();

Правильный ответ: В

Компетенции (индикаторы): ОПК-2.1

4. Какой из предложенных методов сортировки имеет точную асимптотическую оценку:

А) Пузырьковая

Б) Метод выбора

В) Метод вставок

Г) Шейкер-сортировка

Д) Сортировка Шелла

Правильный ответ: Б

Компетенции (индикаторы): ОПК-2.3

*Выберите все правильные варианты ответы*

5. Укажите все типы данных, предназначенные для хранения вещественных значений в языке Си:

А) char

Б) float

В) unsigned long

Г) double

Д) long

Правильные ответы: Б, Г

Компетенции (индикаторы): ОПК-14.3

6. Структуры данных типа LIFO находят применения в задачах:

А) Поиска элемента по ключу

Б) Сохранения состояния при организации вложенных заданий

В) Преобразования выражений в обратную польскую запись

Г) Установления приоритета элементов

Д) Вычисления выражений в обратной польской записи

Е) Организации очередей сообщений

Правильный ответ: Б, В, Д

Компетенции (индикаторы): ОПК-2.1, ОПК-2.3

7. Если операция перемещения элементов является более затратной чем сравнение ключей, то предпочтительно использовать сортировку:

А) Вставками

Б) Выбором

В) Обмена

Г) Индексов

Правильные ответы: Б, Г

Компетенции (индикаторы): ОПК-2.3

8. К наиболее важным областям применения сортировки относятся:

А) Решение задачи группирования по признаку

Б) Вычисление функций агрегации

В) Поиск общих элементов множеств

Г) Цифровая обработка сигналов

Д) Поиск информации по значению ключа

Правильные ответы: А, В, Д

Компетенции (индикаторы): ОПК-2.1, ОПК-2.3

**Задания закрытого типа на установление соответствия**

*Установите правильное соответствие.*

*Каждому элементу левого столбца соответствует только один элемент правого столбца.*

1. Сколько раз выполниться цикл в каждом из случаев, если в теле цикла нет операторов break, return или throw:

|  |  |
| --- | --- |
| 1) for (;;) {} | А) ни одного раза |
| 2) while (0) {} | Б) ровно один раз |
| 3) do {} while(0); | В) конечное число раз (возможно ноль) |
| 4) while (--i) {} | Г) бесконечное число раз |

Правильный ответ: 1-Г, 2-А, 3-Б, 4-В

Компетенции (индикаторы): ОПК-14.3

2. Установите соответствие между спецификаторами формата функции printf и выполняемыми ими преобразованиями:

|  |  |
| --- | --- |
| 1) %5d | А) последовательность символов, начиная с указанного и заканчивающаяся нулевым символом |
| 2) %8.3f | Б) символ заданный своим целым кодом |
| 3) %s | В) целое в шестнадцатиричной системе с ведущими нулями минимум на восьми позициях |
| 4) %p | Г) целое в десятичной системе минимум на пяти позициях |
| 5) %c | Д) длинное целое в десятичной системе |
| 6) %08x | Е) указатель в машинно-зависимой форме |
| 7) %ld | Ж) вещественное минимум на восьми позициях с трема знаками после запятой |

Правильный ответ: 1-Г, 2-Ж, 3-А, 4-Е, 5-Б, 6-В, 7-Д

Компетенции (индикаторы): ОПК-2.1

3. Установите соответствие между синтаксическими конструкциями определения (объявлениями) объектов и их смыслом:

|  |  |
| --- | --- |
| 1) int \*p; | А) массив из десяти структур с двумя целыми полями |
| 2) float\* f(int i, int j); | Б) массив из десяти указателей на функции от вещественного параметра, возвращающих вещественное |
| 3) struct { int x, y; } pt[10]; | В) функция, возвращающая вещественное, определенная в другом месте. |
| 4) double (\*p[10])(double); | Г) указатель на целое |
| 5) double g(); | Д) массив из десяти целых |
| 6) int q[10]; | Е) функция от двух целых параметров, возвращающая указатель на вещественное |

Правильный ответ: 1-Г, 2-Е, 3-А, 4-Б, 5-В, 6-Д

Компетенции (индикаторы): ОПК-14.3

4. Если рассматривать препроцессор как «управляемый сценарием редактор текста», то какие конструкции препроцессора соответствуют каким командам обычного текстового редактора?

|  |  |
| --- | --- |
| 1) #include | А) Найти/Заменить (Find/Replace) |
| 2) #define | Б) Вставка (Paste) |
| 3) #if - #endif | В) Вырезать (Cut) |

Правильный ответ: 1-Б, 2-А, 3-В

Компетенции (индикаторы): ОПК-2.1

5. Установите соответствие между оптимизациями базового алгоритма Хоара и решаемыми ими проблемами быстрой сортировки:

|  |  |
| --- | --- |
| 1) Двустороннее разделение | А) Снижает накладные расходы, связанные с организацией рекурсии, применяя более подходящий алгоритм на практически упорядоченных последовательностях ключей |
| 2) Оптимизация Седжевика (минимальный размер блока) | Б) Ускоряет двустороннее разделение за счет введения барьерных элементов, уменьшает последствия попытки сортировки уже упорядоченных данных. |
| 3) Частичное устранение концевой рекурсии | В) Исключает патологический случай, когда во входных данных один и тот же ключ повторяется многократно. |
| 4) Разделение по медиане трех | Г) Уменьшает глубину стека, если патологии входных данных приводит к выбору опорного элемент далекого от медианы. |

Правильный ответ: 1-В, 2-А, 4-Б, 3-Г

Компетенции (индикаторы): ОПК-2.3

6. Установите соответствие между абстрактными структурами данных свойствами их компонентов:

|  |  |
| --- | --- |
| 1) Однонаправленный список | А) Каждый элемент содержит две ссылки на своих соседей, элементы могут добавляться в любом месте, но «крайние» требуют дополнительного внимания. |
| 2) Двунаправленный список | Б) Элементы содержат две ссылки на своих соседей, имеется специальный псевдо элемент, играющий роль соседа для крайних с обоих сторон, что исключает специальные случаи. |
| 3) Циклический список со сторожевым элементом | В) У каждого элемента по две ссылки на элементы, и имеющие противоположные значения предиката, вычисляемого на основе их ключевых полей. Отличается нелинейной структурой. |
| 4) Двоичное дерево | Г) Элемент содержит указатель на следующий за ним, общее количество элементов может меняться, но эффективны операции только с последним добавленным. |

Правильный ответ: 1-Г, 2-А, 3-Б, 4-Г

Компетенции (индикаторы): ОПК-2.3

7. Установите соответствия между подходами к построению ассоциативных массивов и способами их реализации:

|  |  |
| --- | --- |
| 1) Прямая адресация | А) Сужающая хеш-функция задает не только начальный индекс, но и шаг просмотра, что увеличивает разнообразие возможных последовательностей. |
| 2) Метод цепочек | Б) Сужающая хеш-функция задает один начальный индекс, начина с которого начинается последовательный просмотр ячеек таблицы. |
| 3) Открытая адресация с линейным исследованием | В) Индекс в таблице однозначно вычисляется по ключу, применяется когда количество возможных ключей мало, либо количество используемых ключей близко к полному количеству ключей. |
| 4) Открытая адресация с двойным хешированием | Г) Хеш-функция является сужающим преобразованием, при возникновении коллизии элементы выстраиваются в списки. |

Правильный ответ: 1-В, 2-Г, 3-Б, 4-А

Компетенции (индикаторы): ОПК-2.3

8. Если математическое выражение представлено деревом, в узлах которого находятся операции, а листья являются операндами, установите соответствие между порядком обхода дерева и видом записи выражения, которому он соответствует:

|  |  |
| --- | --- |
| 1) Центральный | А) Префиксная форма записи |
| 2) Прямой | Б) Инфиксная форма записи |
| 3) Обратный | В) Постфиксная форма записи |

Правильный ответ: 1-Б, 3-В. 2-А

Компетенции (индикаторы): ОПК-2.3

**Задания закрытого типа на установление правильной последовательности**

*Установите правильную последовательность. Запишите правильную последовательность букв слева направо.*

1. Из теории автоматического управления известно, что система является устойчивой, если все корни ее характеристического полинома (действительные и комплексные) лежат в левой полуплоскости. Расположите в правильном порядке строки фрагмента кода, реализующего функцию bool stable(float a, float b, float c) {…}, которая проверяет устойчивость системы второго порядка по коэффициентам его полинома, причем a – коэффициент при большей степени, а с – свободный член.

А) return (-b+q)/a < 0 && (-b-q)/a < 0;

Б) double q = sqrt(d);

В) if (d <= 0.0)

Г) double d = b\*b - 4\*a\*c;

Д) return -b/a < 0;

Правильный ответ: Г, В, Д, Б, А

Компетенции (индикаторы): ОПК-2.2, ПК-3.2

2. Расположите в правильном порядке следующие строки, чтобы получился фрагмент программы, считывающий декартовы координаты в виде пар «x, y» и выводящий эти же точки в полярной системе в виде «длина вектора, угол».

А) sqrt(x\*x + y\*y),

Б) while (scanf("%f, %f", &x, &y) == 2)

В) printf("%f, %f\n",

Г) float x, y;

Д) atan2(y, x));

Правильный ответ: Г, Б, В, А, Д

Компетенции (индикаторы): ОПК-2.2

3. Алгоритм обмена с внешним устройством подразумевает, что перед тем как получить блок данных, его надо запросить специальным символом SYN, а после подтвердить получение символом ACK, в том числе для последнего блока, обозначаемого символом EOF. Установите, в каком порядке (слева на право) нужно заполнить пропущенные места в следующем блоке кода:

while ( \_\_ , \_\_ , \_\_ , \_\_ ) { \_\_ }

так чтобы он обеспечивал корректную реализацию алгоритма обмена.

А) data = getc()

Б) /\* обработка принятого блока \*/

В) putc(ACK)

Г) data != EOF

Д) putc(SYN)

Правильный ответ: Д, А, В, Г, Б

Компетенции (индикаторы): ОПК-2.2, ПК-3.3

4. Расположите в правильном порядке стадии подготовки программы и связующие их элементы системы подготовки программ для языка Си:

А) единица трансляции

Б) редактор связей

В) компилятор

Г) исполняемый код

Д) препроцессор

Е) объектный код

Ж) исходный код

Правильный ответ: Ж, Д, А, В, Е, Б, Г

Компетенции (индикаторы): ОПК-14.1

5. Расположите асимптотические оценки по возрастанию сложности:

А) O(N log N)

Б) O(2^N)

В) O(1)

Г) O(N^2)

Д) O(log N)

Правильный ответ: В, Д, А, Г, Б

Компетенции (индикаторы): ОПК-2.3

6. Расположите элементы дерева в порядке, соответствующим прямому обходу:

А) Элементы правого поддерева, при наличии

Б) Сам элемент

В) Элементы левого поддерева, при наличии

Правильный ответ: Б, В, А

Компетенции (индикаторы): ОПК-2.3

7. Пусть задан алгоритм “Разделение(A,p,r) ”, который производит разделение по Ламутто массива A в диапазоне индексов от p до r и возвращает индекс опорного элемента. Расположите нижеприведенные операции псевдокода в порядке, позволяющем получить тело цикла алгоритма поиска i-й порядковой статистики:

А) p = q+1, i = i – n

Б) ИНАЧЕ

В) n = q – p + 1

Г) ЕСЛИ n < i ТО

Д) ЕНАЧЕ ЕСЛИ n > i ТО

Е) q = Разделение(A,p,r)

Ж) ВЕРНУТЬ A[q]

З) r = q – 1

Правильный ответ: Е, В, Г, А, Д, З, Б, Ж

Компетенции (индикаторы): ОПК-2.3, ОПК-2.2

8. Расположите в правильном порядке нижеприведенные операторы псевдокода, чтобы получить алгоритм коррекции свойства порядка невозрастающей кучи A, текущий размер который равен N, нарушенного в результате добавления элемента:

А) ЦИКЛ ПОКА i > 1

Б) ЕСЛИ A[j] > A[i] ТО

В) ИНАЧЕ

Г) ПРЕРВАТЬ ЦИКЛ

Д) i = N

Е) j = i / 2

Ж) ОБМЕНЯТЬ(A[i], A[j]), i = j

Правильный ответ: Д, А, Е, Б, Ж, В, Г

Компетенции (индикаторы): ОПК-2.3, ОПК-2.2

**Задания открытого типа**

**Задания открытого типа на дополнение**

*Напишите пропущенное слово (словосочетание).*

1. Ситуация, когда две функции в процессе вычисления результата или выполнения действия могут обращаться одна к другой, называется \_\_\_\_\_\_\_\_\_.

Правильный ответ: взаимная рекурсия

Компетенции (индикаторы): ОПК-14.1

2. Совокупность текста, получающаяся после выполнения всех директив препроцессора, таких как включение других источников, условное исключение блоков и операции замены и подстановки текста, называется \_\_\_\_\_\_\_\_\_\_.

Правильный ответ: единица трансляции

Компетенции (индикаторы): ОПК-14.1

3. Отличительной особенностью логических версий операций AND и OR в языке Си, позволяющих писать более компактный код, не прибегая к вложенным операторам if, является то, что они вычисляются по \_\_\_\_\_\_\_\_\_\_.

Правильный ответ: короткой схеме

Компетенции (индикаторы): ОПК-14.1

4. В тех случаях, когда синтаксис языка Си предполагает единственное выражение, а логика программы требует несколько действий, выполняемых безусловно и в определенной последовательности, следует использовать операцию \_\_\_\_\_\_\_\_\_\_.

Правильный ответ: следования

Компетенции (индикаторы): ОПК-14.1

5. Метод сортировки называется \_\_\_\_\_\_\_\_\_\_\_\_\_, если в процессе сортировки относительное расположение элементов с равными ключами не меняется.

Правильный ответ: устойчивым

Компетенции (индикаторы): ОПК-2.3

6. Когда наименьшее время выполнения соответствует уже отсортированным данным, а наихудше – случаю, когда данные отсортированы в обратном порядке, говорят, что сортировка демонстрирует \_\_\_\_\_\_\_\_\_\_\_\_\_\_.

Правильный ответ: естественное поведение

Компетенции (индикаторы): ОПК-2.3

7. Двоичное дерево, на всех уровнях *i* которого, кроме, возможно, последнего, присутствуют максимальное количество узлов, равное *2i*, при этом может отсутствовать правая часть узлов последнего уровня, называется \_\_\_\_\_\_\_\_\_\_\_.

Правильный ответ: почти полным

Компетенции (индикаторы): ОПК-2.3

8. Абстрактная структура данных, определяемая через две операции «вставить» и «удалить», соответственно добавляющие и удаляющие элементы в порядке, определяемом самой структурой таким образом, что удаляется всегда наименьший (или наибольший) элемент, называется \_\_\_\_\_\_\_\_\_\_\_.

Правильный ответ: очередь с приоритетом

Компетенции (индикаторы): ОПК-2.3

**Задания открытого типа с кратким свободным ответом**

*Дайте ответ на вопрос.*

1. Запищите команду, позволяющую выполнить компиляцию отдельно транслируемого блока кода, расположенного в файле src.c, с помещением объектного кода в файл module.o средствами компилятора GCC.

Правильный ответ: gcc -c -o module.o src.c

Компетенции (индикаторы): ОПК-14.3

2. Программа в файле test1.c использует библиотеку, представленную заголовочным файлом rccl.h и архивами объектных модулей в файлах librcclcore.a и librcclext.a, расположенных по путям, известным компилятору и редактору связей. Напишите команду для сборки этой программы в случае использования компилятора GCC с получением исполняемого модуля с именем example.

Правильный ответ: gcc -o example test1.c -lrcclcore -lrcclext

Компетенции (индикаторы): ОПК-14.2

3. Дан массив вида ‘double data[4][4]’. Инициализируйте все элементы значением Z, используя единственный цикл for и арифметику указателей.

Правильный ответ: for (double \*s = &data[0][0], \*e = s + 4\*4; s < e; ++s) \*s = Z;

Компетенции (индикаторы): ОПК-2.2

4. В текстовом файле двумерные отрезки представлены указанием начальных и конечных координат в формате с плавающей точкой взятых в скобки и разделенных точкой с запятой, между скобками стоит символ минус, например "(1.1;2.0)-(3.3;4.0)". При этом допускается наличие пробельных символов вне числовых литералов, например "(5.1; 2.0 ) -(3.3 ;4.0 ) ". Запишите строку формата \*scanf функции для сохранения четырех значений в объектах типа double. При записи ответа на бумаге явно указывайте положение пробелов символом ‿.

Правильный ответ: "‿ (‿%lf‿,‿%lf‿)‿-‿ (‿%lf‿,‿%lf‿)".

Компетенции (индикаторы): ОПК-2.3; ОПК-14.3;

5. Свойство формы кучи (heap) позволяет вычислять индекс правого дочернего узла *i*-го узла в представляющем массиве по формуле \_\_\_\_\_\_\_\_\_.

Правильный ответ: 2*i* + 1

Компетенции (индикаторы): ОПК-2.3

*Напишите результат вычислений.*

6. Анализ алгоритма показал, что время его выполнения пропорционально (2\*N+1)(N + log N)+3N, где N – количество объектов во входных данных. Чему равна верхняя асимптотическая оценка данного алгоритма?

Правильный ответ: O(N^2)

Компетенции (индикаторы): ОПК-2.3

7. Как известно, основное улучшение Кнутом сортировки Шелла заключалось в выборе шага по формуле h0=1, hS+1 = 3hS +1. Пусть в вычислительной системе размер адресного пространства ограничен 4Гб. Грубо оцените верхнюю границу количества шагов разбиения по Кнуту.

Правильный ответ: N < log3232 = 32 log32 < 32

Компетенции (индикаторы): ОПК-2.3

8. Преобразуйте выражение (3x^2+1)(2x-5y) в обратную польскую запись, используя символ | для разделения символов (операция push).

Правильный ответ: x | ^2 | 3 | \* | 1 | + | 2 | x | \* | 5 | y | \* | - | \*

Компетенции (индикаторы): ОПК-2.3

**Задания открытого типа с развернутым ответом**

1. Запишите функцию, заданную прототипом:

 double inverse(double r, double l, double x);

решающую с применением теоремы косинусов обратную задачу кинематики для кривошипно-шатунного механизма:

исходя из предположения, что r < l конструктивно. Поясните вывод.

Время выполнения – 35 мин.

Ожидаемый результат:

 Согласно теореме косинусов l2 = r2 + x2 – 2‧r‧x‧cos(), откуда 2‧r‧x‧cos() = r2 + x2 - l2, и  = arccos((r2 + x2 - l2)/(2‧r‧x)). Т.к. по условию r < l, то конструктивно x≠0, получаем:

double inverse(double r, double l, double x) {

 return acos((r\*r + x\*x - l\*l)/(2\*r\*x));

}

Критерии оценивания:

 - студент знаком с элементами мехатронных и робототехнических систем;

 - студент владеет математическим аппаратом на уровне достаточном для описания простейших механизмов;

 - студен способен корректно выражать математические зависимости средствами языков высокого уровня

Компетенции (индикаторы): ОПК-2.2; ОПК-14.3; ПК-3.2

2. Задана структура:

typedef struct { float x, y, z; } POINT;

Используя библиотечную функцию qsort() и вспомогательную функцию сравнения, локальную на уровне модуля, напишите функцию упорядочивающую вектор POINT-ов в порядке удаления его компонентов от начала системы координат, заданную прототипом:

 void point\_sort(POINT points[], int count);

Время выполнения – 35 мин.

Ожидаемый результат:

static int compare(const void \*left, const void \*right) {

 const POINT \*pl = (const POINT\*)left,

 \*pr = (const POINT\*)right;

 float delta = (pl->x\*pl->x + pl->y\*pl->y + pl->z\*pl->z) -

 (pr->x\*pr->x + pr->y\*pr->y + pr->z\*pr->z);

 return delta < 0 ? -1 : (delta > 0 ? +1 : 0);

}

void point\_sort(POINT points[], int count){

 qsort(points, count, sizeof(POINT), compare);

}

Критерии оценивания: правильный ответ должен включать вспомогательную функцию, определенную с модификатором static для сравнения двух объектов POINT согласно условию задачи и заданных в обобщенном виде; корректно обрабатывать переход от вещественного результата геометрических вычислений к целочисленному интерфейсу функции сравнения; корректное обращение к библиотечной функции qsort().

Компетенции (индикаторы): ОПК-2.2; ОПК-2.3; ОПК-14.2;

3. Как известно, в языках C/C++ матрица A[N][M] может быть представлена указателем на свой начальный элемент &A[0][0], при этом в памяти она храниться как последовательность из N строк по M элементов в каждой. Напишите функцию:

void mmul(double \*A, double \*B, double \*C, int L, int M, int N),

вычисляющую результат умножения матриц:

A(LxM) \* B(MxN) -> C(LxN).

Время выполнения – 40 мин.

Ожидаемый результат:

void mmul(double \*A, double \*B, double \*C, int L, int M, int N){

 for (int i = 0; i < L; ++i) {

 for (int j = 0; j < N; ++j) {

 double s = 0;

 for (int k = 0; k < M; ++k)

 s += A[i\*M + k] \* B[k\*N + j];

 C[i\*N + j] = s;

 }

 }

}

Критерии оценивания:

 - корректность адресации элементов матриц с учетом их геометрии;

 - корректность реализации математического алгоритма

Компетенции (индикаторы): ОПК-2.2

4. Манипулятор, работающий в плоскости ZY и состоящий из вращательных (R) и поступательных (T) звеньев может быть представлен структурной схемой S и вектором конструктивных параметров P, а его текущее положение – вектором обобщенных координат C. При этом для вращательных звеньев P[i] = const длина звена, C[i] – угол поворота по отношению к предыдущему звену. Для поступательных звеньев P[i] = const угол монтажа, C[i] – перемещение звена. Компоненты структурной схемы описывают типы звена 'R' или 'T', либо содержат вспомогательную информацию. В следующем примере S = "R-T", P = { b, c }, C = {a, d}:

Напишите функцию для решения прямой задачи кинематики, заданную следующим прототипом:

void direct(float \*y, float \*z, char \*S, float \*P, float \*C);

Время выполнения – 55 мин.

Ожидаемый результат:

void direct(float \*y, float \*z, char \*S, float \*P, float \*C)

{

 double rotation = 0;

 for (\*y = \*z = 0; \*S; ++S) {

 switch (\*S) {

 case 'R':

 rotation += \*C;

 \*y += \*P \* cos(rotation);

 \*z += \*P \* sin(rotation);

 break;

 case 'T':

 rotation += \*P;

 \*y += \*C \* cos(rotation);

 \*z += \*C \* sin(rotation);

 break;

 default:

 continue;

 }

 ++P, ++C;

 }

}

Критерии оценивания: правильный ответ должен учитывать суммирование углов поворота и суммирование переносов вдоль осей, вносимых каждым последующим звеном; корректно интерпретировать компоненты векторов параметров и обобщенных координат в зависимости от типа звена; корректно обрабатывать содержание в строке структурной схемы вспомогательных символов.

Компетенции (индикаторы): ОПК-2; ОПК-14; ПК-3.2

5. Рассмотрите трансформацию рабочего органа манипулятора в двумерном случае, когда возможны вращения вокруг оси X и перемещение вдоль осей Y и Z, при этом сам рабочий орган описывается четырьмя числами: перенос по X и Y, и проекции нормированного вектора ориентации на оси X и Y.

 Создайте структуру, описывающую матрицу трансформации для этого случая, предусмотрев два конструктора. В первом случае параметры конструктора задают тип матрицы трансформации Rx, Ty, Tz и E – единичная, и параметр – величина угла поворота или переноса. Во втором случае новая матрица строится как результат перемножения двух других матриц трансформации.

 С помощью созданной структуры опишите манипулятор Rx(45)-Ty(100)-Tz(100).

Время выполнения – 60 мин.

Ожидаемый результат:

struct Transform {

 typedef enum { Rx, Ty, Tz, E } type;

 Transform(type t, double p) {

 for (int i = 0; i < 4; ++i)

 for (int j = 0; j < 4; ++j)

 data[i][j] = (i == j) ? 1.0 : 0.0;

 switch (t) {

 case Rx: {

 double rad = p \* 3.1415926 / 180;

 double c = cos(rad), s = sin(rad);

 data[2][2] = c; data[2][3] = -s;

 data[3][2] = s; data[3][3] = c;

 break;

 }

 case Ty:

 data[0][2] = p; data[1][3] = p;

 break;

 case Tz:

 data[0][3] = -p; data[1][2] = p;

 break;

 }

 }

 Transform(const Transform& left, const Transform& right) {

 for (int i = 0; i < 4; ++i)

 for (int j = 0; j < 4; ++j) {

 double s = 0;

 for (int k = 0; k < 4; ++k) s += left.data[i][k] \* right.data[k][j];

 data[i][j] = s;

 }

 }

 double data[4][4];

};

void example() {

 Transform link1(Transform::Rx, 45.0),

 link2(Transform::Ty, 100),

 link3(Transform::Tz, 100);

 Transform manipulator(Transform(l3, l2), l1);

}

Критерии оценивания:

 - присутствует идентификация типа трансформации, например в виде перечисления

 - конструктор 1 корректно инициализирует данные для четырех типов трансформации

 - конструктор 2 корректно вычисляет произведение матриц

 - созданы три звена, с параметрами, соответствующими заданию

 - при получении трансформация всего манипулятора правильно учтена структура манипулятора и т.о. порядок перемножения матриц.

Компетенции (индикаторы): ОПК-2.3; ОПК-14.2; ОПК-14.3;ПК-3.3

6. В программе моделирования робототехнической системы было решено представить манипулятор в виде однонаправленного связного списка, в элементах которого хранится информация о типе звена и его параметр (величина угла поворота или перемещение). В следующем примере должен создаваться манипулятор и печататься его структура в виде строки «Rx(45)-Ty(100)-Tz(100)».

#include <iostream>

struct Link {

 enum { Rx, Ty, Tz } type;

 float value;

 Link \*next;

};

int main() {

 Link l1 = { Link::Rx, 45 }, l2 = { Link::Ty, 100 }, l3 = { Link::Tz, 100 };

 Link &arm = l1 + l2 + l3;

 std::cout << arm << std::endl << std::endl;

}

Определите, каких элементов не хватает в примере, и приведите их исходный код.

Время выполнения – 60 мин.

Ожидаемый результат:

Link& operator+(Link& left, Link &right) {

 Link \*p = &left;

 while (p->next) p = p->next;

 p->next = &right;

 return left;

}

std::ostream& operator<<(std::ostream& os, const Link& link) {

 for (const Link \*p = &link; ; p = p->next) {

 switch (p->type) {

 case Link::Rx: os << "Rx("; break;

 case Link::Ty: os << "Ty("; break;

 case Link::Tz: os << "Tz("; break;

 }

 os << p->value << ")";

 if (!p->next)

 break;

 os << "-";

 }

 return os;

}

Критерии оценивания:

 - определены перегруженные версии операторов '+' и '<<' для корректных типов данных;

 - алгоритм оператора '+' обеспечивает помещение элементов в список в ожидаемом порядке;

 - возвращаемое оператором '+' значение обеспечивает корректное присвоение в программе-примере;

 - алгоритм оператора '<<' обеспечивает корректный порядок обхода и печати списка звеньев.

Компетенции (индикаторы): ОПК-2.3; ОПК-14.2; ОПК-14.3;ПК-3.3

7. Пусть в программе моделирования робототехнической системы манипулятор описывается связанным списком с элементами типа Link, а положение рабочего органа типом Point, назначение полей которых указаны в комментариях:

struct Link {

 enum { Rx, Ty, Tz } type; // вращение вокруг X, перемещение по Y или Z

 float value; // угол поворота (градусы) или перемещение (мм)

 Link \*next; // следующее звено

};

struct Point {

 float y, z, // координата на плоскости YZ

vy, vz; // нормированная проекция вектора направление на оси Y и Z

};

Напишите код, позволяющий скомпилировать следующую строку:

Point q = p \* arm;

где:

arm – манипулятор, заданный своим первым звеном;

p – точка, в которой находится манипулятор;

q – точка, в которой находится рабочий орган.

Время выполнения – 60 мин.

Ожидаемый результат:

Point operator\*(const Point& pt, const Link& list) {

 Point r = pt;

 for (const Link \*p = &list; p; p = p->next) {

 switch (p->type) {

 case Link::Rx: {

 double rad = p->value \* 3.1415926 / 180;

 double c = cos(rad);

 double s = sin(rad);

 double y = r.vy \* c - r.vz \* s;

 double z = r.vy \* s + r.vz \* c;

 r.vy = y;

 r.vz = z;

 break;

 }

 case Link::Ty:

 r.y += r.vy \* p->value;

 r.z += r.vz \* p->value;

 break;

 case Link::Tz:

 r.y += -r.vz \* p->value;

 r.z += r.vy \* p->value;

 break;

 }

 }

 return r;

}

Критерии оценивания:

 - определена перегруженная версия оператора '\*', правильно заданы типы параметров и возвращаемый тип, так что оператор порождает новую точку;

 - корректно реализован алгоритм обхода звеньев манипулятора;

 - корректно идентифицируется тип звена, и выполняются соответствующие вычисления.

Компетенции (индикаторы): ОПК-2.3; ОПК-14.2; ОПК-14.3;ПК-3.3

8. Даны две структуры, описывающие положение рабочего органа в пространстве и базовый элемент списка, представляющего манипулятор как последовательность звеньев. Значение полей указано в комментариях:

struct Point {

 float y, z, // координата на плоскости YZ

vy, vz; // нормированная проекция вектора направление на оси Y и Z

};

struct Link {

 Link(double v) : value(v), next(0) {}

 virtual void translate(Point &pt) const { if (next) next->translate(pt); }

 double value; // угол поворота вокруг X в градусах или перемещение в мм.

 Link \*next; // следующее звено

};

Опишите наследники Link - Rx, Ty и Tz описывающие конкретные кинематические пары, переопределяя операцию translate и оператор, позволяющий скомпилировать следующий код:

Point q = p \* arm;

где:

arm – манипулятор, заданный своим первым звеном;

p – точка, в которой находится манипулятор;

q – точка, в которой находится рабочий орган.

Время выполнения – 60 мин.

Ожидаемый результат:

struct Rx : Link {

 Rx(double v) : Link(v) {}

 void translate(Point &pt) const {

 double rad = value \* 3.1415926 / 180;

 double c = cos(rad), s = sin(rad);

 double vy = c \* pt.vy - s \* pt.vz;

 double vz = s \* pt.vy + c \* pt.vz;

 pt.vy = vy;

 pt.vz = vz;

 Link::translate(pt);

 }

};

struct Ty : Link {

 Ty(double v) : Link(v) {}

 void translate(Point &pt) const {

 pt.y += value \* pt.vy;

 pt.z += value \* pt.vz;

 Link::translate(pt);

 }

};

struct Tz : Link {

 Tz(double v) : Link(v) {}

 void translate(Point &pt) const {

 pt.y += -value \* pt.vz;

 pt.z += value \* pt.vy;

 Link::translate(pt);

 }

};

Point operator\*(const Point& pt, const Link& link) {

 Point r = pt;

 link.translate(r);

 return r;

}

Критерии оценивания:

 - определена перегруженная версия оператора '\*', правильно заданы типы параметров и возвращаемый тип, так что оператор порождает новую точку;

 - обхода звеньев манипулятора реализован через возможность, заложенную в базе Link;

 - в типах кинематических пар корректно определена соответствующая процедура вычисления преобразования координат.

Компетенции (индикаторы): ОПК-2.3; ОПК-14.2; ОПК-14.3;ПК-3.3