Задачи:

Часть A – на оценку хор

1. Структуры данных
   Что нужно: Знать что такое O(n) нотация и уметь определять сложность алгоритма. Знать что такое список, дерево и хэш таблица, как они реализуются в языке C. Знать сложности операций добавления, удаления, взятия элемента по индексу и поиска элемента в этих структурах данных.
   Пример задачи: Объяснить реализацию хэш-таблицы в языке C.

2. Битовые операции.
   Что нужно: Знать что такое побитовые операторы: побитовое не, побитовое или, побитовое и, исключающее или. Знать что такое битовые сдвиги (логический и арифметический).
   Пример задачи: Используя битовые операции умножить целое число на 128. Используя битовые операции найти остаток деления целого числа на 2^n. Используя битовые операции найти значение k-го бита целого числа.

3. Представление целых чисел в памяти компьютера.
   Что нужно: Знать как представляются целые числа в памяти компьютера. Отдельно рассмотреть беззнаковые целые числа и знаковые(дополнительный код). Little endian и big endian.
   Пример задачи: Как представляется в памяти компьютера число -5000?

4. Представление чисел с плавающей точкой в памяти компьютера.
   Что нужно: Знать как представляются числа с плавающей точкой в памяти компьютера. Знать как представляются особые значения inf, nan, денормализованные числа. Little endian и big endian.
   Пример задачи: Как представляется в памяти компьютера число -2.0?

5. Трансляция C кода в объектный код.
   Что нужно: Знать все стадии трансляции C кода в объектный код(Препроцессинг, компиляция кода на языке C в код на языке ассемблера, компиляция ассемблерного кода в машинный код, линковка.). Уметь использовать gcc для исполнения этих стадий трансляции кода.

6. Программная модель ассемблера. Основы языка ассемблера.
   Что нужно: Знать что такое центральный процессор, регистры, счётчик команд, коды условий. Знать чем различаются наборы регистров для 32-битных систем и для 64-битных. Операция mov и lea. Адресация памяти.
   Пример задачи: Пусть в регистрах eax и ebx лежат значения 0x15 и 0xf соответственно. Что будет лежать в регистре ecx после выполнения следующей операции

   ```
   leal 0x500(%eax, %ebx, 4), %ecx
   ```

7. Язык ассемблера.
   Что нужно: Арифметические команды: add, sub, imul, idiv, sal, sar, xor, and, or, inc, dec. Флаги условий(CF, ZF, SF, OF), как эти флаги устанавливаются и используются. Команды cmp, test, set. Условные переходы. Команды j*(jmp, jg, jl и т. д.). Реализация циклов в языке ассемблера с помощью условных переходов. Вызов функции в языке ассемблера.
   Пример задачи: Написать программу на языке ассемблера, которая считывает 2 целых положительных числа и возводит первое число в степень другого.

Часть B – на оценку отл

1. Машина Тьюринга.
   Что нужно: Знать что такое машина Тьюринга и зачем она нужна. Что такое детерминированная и недетерминированная машины Тьюринага. Классы сложности задач: P, NP, NP complete. Привести примеры задач, которые:

   (a) входят в класс P

   (b) входят в класс NP, но не входят в P

   (c) входят в класс NP complete

   (d) не входят ни в один из перечисленных классов, но не являются невычислимыми

   (e) невычислимы

2. Графы.
   Что нужно: Знать что такое граф, как он может быть представлен (матрица смежности и список смежных вершин) и преемущества/недостатки этих представлений. Различные виды графов: связный граф, взвешенный граф, ациклический граф, дерево. Алгоритмы поиска в глубину и поиска в ширину в графе. Абстрактный тип данных очередь с приоритетом, реализация этого абстрактного типа данных с помощью структуры данных heap(куча). Алгоритм Дейкстры и использование очереди с приоритетом в этом алгоритме.

3. Стек.
   Что нужно: Знать что такое стек и как он используется в языке C и в языке ассемблера. Команды push и pop. Особый регистр rsp(или esp в 32-битных системах). Вызов процедуры в языке ассемблера, адресс возврата, команды call и ret. Передача аргументов в процедуру и возращаемый результат. Использование регистров в x86-64.
   Пример задачи: Написать рекурсивную функцию вычисления факториала на языке ассемблера.

4. Иерархия памяти. Кэш.
   Что нужно: Принцип локальности. Иерархия памяти. Что такое регистры, кэш(несколько уровней), основная память, локальные диски, удалённые хранилища. Общие принципы устройства кэша: попадания кэша, промахи кэша(холодный промах, конфликтный промах, промах ёмкости), линия кэша. Характерные значения вероятности промахов и продолжительности доступа в кэш и в основную память в современных архитектурах.
   Пример задачи: Вычислить процент промахов в следующем участке кода. Длина кэш-линии – 64 байта. Размер кэша – 1 мегабайт.

```
#define N 10000
...
int A[N], B[2*N], sum=0;
...
for (int i = 0; i < N; ++i) {
    sum += A[i] + B[2*i];
}
```