Ворожцов Артём Викторович
Кандидат физико-математических наук,
преподаватель кафедры информатики
Московского физико-технического института (МФТИ),
тренер сборной команды МФТИ
по программированию.

Тестирование по информатике. Часть 1.

Этот тест посвящён алгоритмам и парадигмам программирования. Многие вопросы и варианты ответов имеют шутливый характер, но среди них есть довольно сложные. Правильный выбор варианта часто будет требовать глубокого понимания темы. В конце теста приведены замечания и правильные ответы. Сразу отметим, что для некоторых вопросов дано несколько правильных ответов, для некоторых – ни одного (все варианты не совсем правильные), но большинство вопросов имеют ровно один правильный ответ.

1. Алгоритмы бывают
a) однопроходные;
b) вперёд-и-назад-проходные;
c) двупроходные;
d) непроходимые.

2. Алгоритм Евклида вычисляет
a) площадь прямоугольника;
b) площадь равнобедренного треугольника;
c) наибольший общий делитель;
d) вашу оценку по информатике;
e) самое большое простое число.

3. Метод деления пополам – это
a) когда один делит, а другой выбирает;
b) способ разделить пирог (или верёвку) на 4, 8, 16 и вообще на 2^n частей за n разрезов;
c) метод, в котором на каждом шаге область поиска уменьшается в два раза;
d) один из ужасных типов казни еретиков в Италии в средние века.

4. Рекурсия – это
- a) сведение вычисления значения функции к вычислению значений этой же функции при других значениях аргументов;
- b) разбиение задачи на отдельные маленькие подзадачи, которые выполняются пошагово;
- c) метод построения программы, в котором одни функции используют другие, те, в свою очередь, используют третьи и так далее до элементарных функций;
- d) это то, что использовать крайне не рекомендуется, так как именно рекурсия приводит к зависанию компьютеров: какая-нибудь функция начинает бесконечно сама себя вызывать, вызывать, вызывать, вызывать, вызывать...

5. Ленивые вычисления – это
- a) вычисления на низко производительных компьютерах;
- b) вычисления в режиме отладки, которые осуществляются в замедленном темпе;
- c) вычисления, которые откладывают на потом;
- d) такие вычисления, которые двигают прогресс.

6. Кто ленивее: транслятор программы на функциональном языке программирования или программист, который пишет эту программу?
- a) конечно, программист!
- b) конечно, транслятор!
- c) оба друг друга стоят!

7. Динамическое программирование – это
- a) когда память, необходимая для хранения данных, выделяется во время выполнения программы;
- b) когда параметризуют множество задач и постепенно расширяют множество решённых задач, начиная с простых;
- c) когда активно используют двумерные и трёхмерные массивы, чтобы хранить промежуточные данные;
- d) когда выигрывают в скорости работы алгоритма за счёт активного использования памяти.

8. Жадные алгоритмы – это
- a) алгоритмы, которые построены на основе локально жадной стратегии: выбирай, что есть побольше и получше, и в результате получишь оптимум;
- b) алгоритмы зарабатывания денег на биржах ценных бумаг, основанные на прогнозировании с помощью генетических алгоритмов и естественного отбора;
- c) алгоритмы, которые захватывают системные ресурсы и приводят к глобальному «торможению» и «зависанию» системы;
- d) алгоритмы, которые работают неполиномиальное от размера входа время.

9. Время жизни переменной – это
a) время от момента, когда её придумал программист, до момента, когда он ею перестал пользоваться;
b) время, пока эта переменная располагается в памяти процесса во время его исполнения;
c) время до уничтожения переменной компилятором языка;
d) время до уничтожения переменной программистом.

10. Динамическое выделение памяти – это
a) динамическое отделение блоков оперативной памяти от материнской платы компьютера;
b) активизация бездействующих чипов оперативной памяти по мере надобности;
c) автоматическое расширение оперативной памяти компьютера во время его работы;
d) запрос к ядру операционной системы предоставить программе (процессу) память.

11. Императивный программист – это программист,
a) которого нужно срочно увольнять;
b) который делает основную работу за ленивых функциональных программистов;
c) который думает о программе как о последовательности действий с условными и безусловными переходами;
d) который думает о программе как о множестве определений функций, зависящих друг от друга;
e) который не думает, а делает;
f) который использует оператор `goto`.

12. Когда следует использовать функциональную парадигму программирования?
a) когда преподаватель того требует;
b) всегда, так как она более удобная;
c) никогда, поскольку она всегда менее эффективна, нежели обычная парадигма структурного императивного программирования;
d) когда не важно время работы программы, а время программиста (ваше время) дорого;
e) когда задача такова, что она естественно ложится на функциональную парадигму.

13. Какой этап разработки программного обеспечения наиболее важен?
a) придумывание базовой идеи;
b) проектирование;
c) прототипирование;
d) тестирование и отладка;
e) разговоры во время обеденного перерыва с коллегами;
f) маркетинг и продажа;
g) отпуск на Кипре;
h) получение денег от заказчика.

14. Reuse (повторное использование) – это
- a) метод построения сложных систем, основанный на активном использовании универсальных общих инструментов, и новые системы разрабатываются так, чтобы результат также был представлен в максимально удобном и общем виде;
- b) когда код, разработанный, но заброшенный и забытый одними программистами, дорабатывается и повторно используется другими, а потом тоже забрасывается;
- c) один из этапов разработки программного обеспечения, в котором то, что разрабатывалось в прошлом цикле, в этот раз снова используется;
- d) метод построения новых систем из существующих «кирпичиков», готовых инструментов, библиотек, других систем.

15. Интерфейс библиотеки функций – это
- a) то, как внешне выглядит библиотека, в которой хранятся списки математических функций;
- b) способ взаимодействия приложений друг с другом;
- c) внутреннее устройство библиотеки функций, то есть описание того, как функции зависят друг от друга;
- d) термин, который произошёл от английского “into face”, то есть “внутрь лица”, то есть “посмотреть в глаза библиотеке функций”, то есть это средство посмотреть в глаза разработчику данной библиотеки, а значит – это исходный код библиотеки функций с описанием всех функций;
- e) список функций с описанием их сигнатуры (как называются, что получают на вход и что возвращают в качестве результата) и семантики (то есть назначения функции – что она, собственно, делает).

Правильные ответы и замечания

1.a, c; 2.c; 3.c; 4.a; 5.c; 6.c; 7.b, а также косвенными признаками являются c и d; 8.a; 9.b; 10.d; 11.d (кстати, пункт e относится к функциональному программисту); 12.e; 13.b, d (маркетинг и продажа, отпуск на Кипре, получение денег от заказчика, безусловно, являются важными этапами в жизни разработчика программного обеспечения, но они не являются этапами разработки); 14.a, d; 15.e.

Некоторые ответы к тестам имеют неоднозначный характер. Если у Вас возникли вопросы или замечания относительно правильного варианта ответа, то обращайтесь по адресу info@potential.org.ru. Мы с удовольствием ответим на Ваши вопросы и замечания. Кроме того, если Вы – учитель информатики, и у Вас есть собственные интересные подборки тестов по информатике «на понимание», то присылайте его по тому же адресу с темой «тесты по информатике».