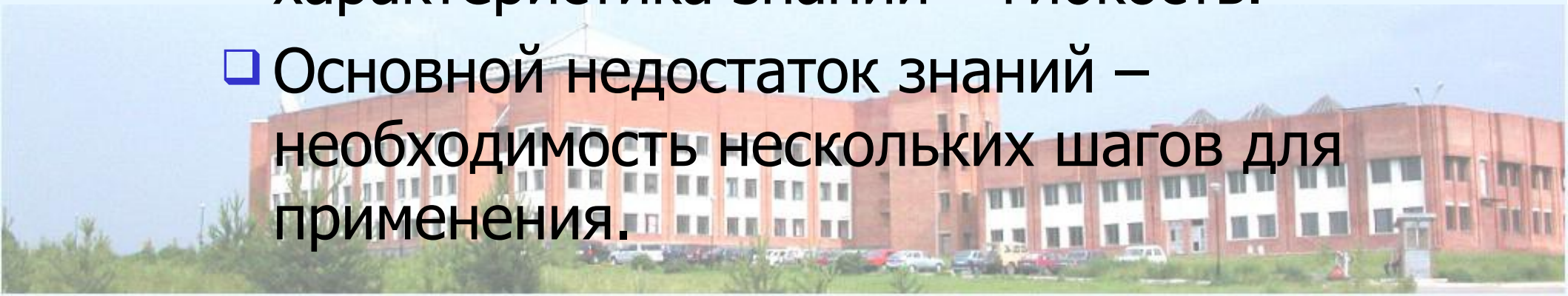
О необходимых знаниях и умениях для программистов суперкомпьютеров Н. Н. Непейвода

Основные понятия

- Данные – хранятся, извлекаются по запросу, модифицируются.
- Умения – применяются.
- Знания – модифицируются, порождают данные, знания и умения.
- Таким образом, основная характеристика знаний – гибкость.
- Основной недостаток знаний – необходимость нескольких шагов для применения.

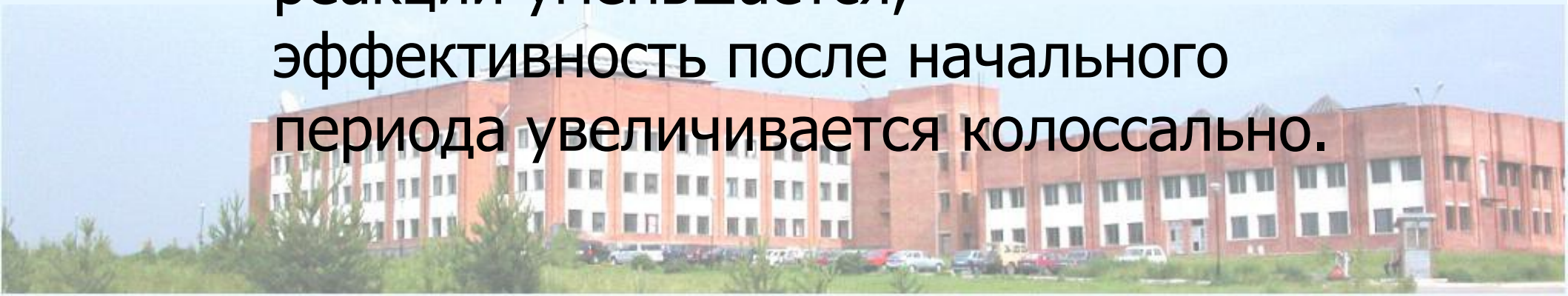


Маски

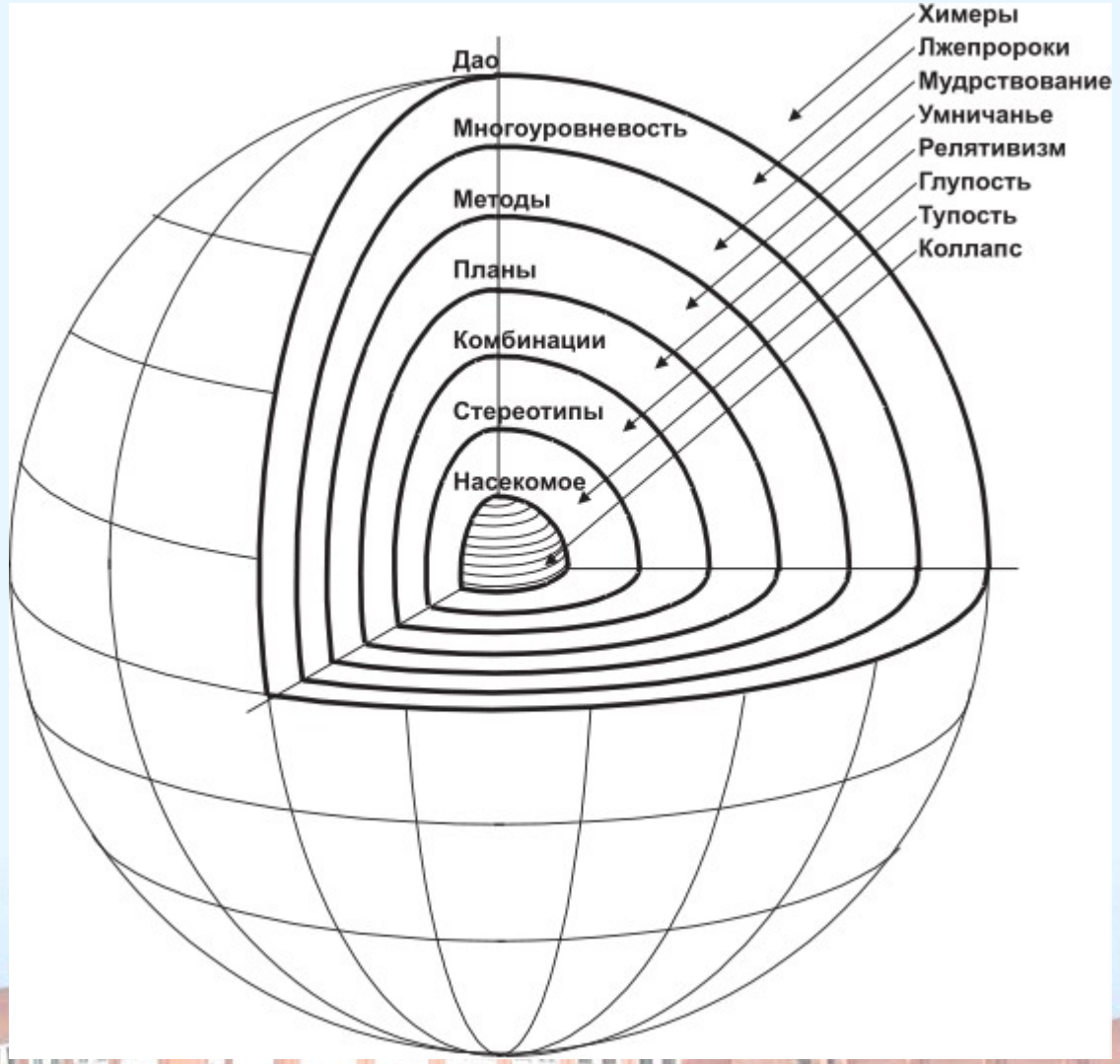
Что-как Плюсы- минусы	данные	умения	знания
данные		Навык + производительность - Гибкость	Эрудиция ----
умения	Наставление, устав, программа +Система - Ограниченность.		Мастерство, ++++
знания	Учебник -----	Прикладная наука, + +++	

Уровни

- Знания и умения различаются уровнем.
- Высший уровень порождает низшие.
- Чем выше уровень, тем труднее применить конкретно и тем легче найти новое решение.
- Гибкость увеличивается, быстрота реакции уменьшается, эффективность после начального периода увеличивается колоссально.



Уровни 2


Химеры
Лжепророки
Мудрствование
Умничанье
Релятивизм
Глупость
Тупость
Коллапс
Дао
Многоуровневость
Методы
Планы
Комбинации
Стереотипы
Насекомое

Уровни 3

- Чем выше уровень, тем меньше людей, способных его понять (уменьшение в 10-100 раз)
- Чем ниже уровень, тем быстрее реакция и легче объяснить решение
- Чем выше уровень, тем быстрее переход к новому (оно оказывается частным случаем высокоуровневого старого)
- Сферы знания тонкие, пространства невежества широки

Уровни 4

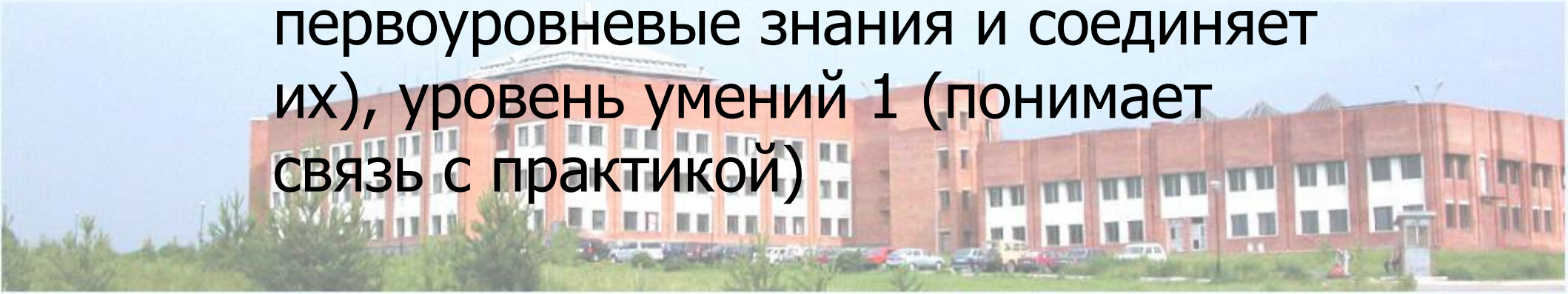
- Достаточно точная характеризация уровней: тип объектов, с которыми может работать человек
- Первый уровень: функции и свойства
- Второй: объекты, функции над функциями, свойства свойств
- Уровень метода: примерно 4
- Мало кто добирается, исключительно устойчивый и плодотворный

Уровни 5

- Человек может иметь разные уровни в разных областях.
- Гармоничное сочетание – когда в данной области уровни знаний и умений различаются на 1
- На соседних уровнях достоинства и недостатки, как правило, меняются местами

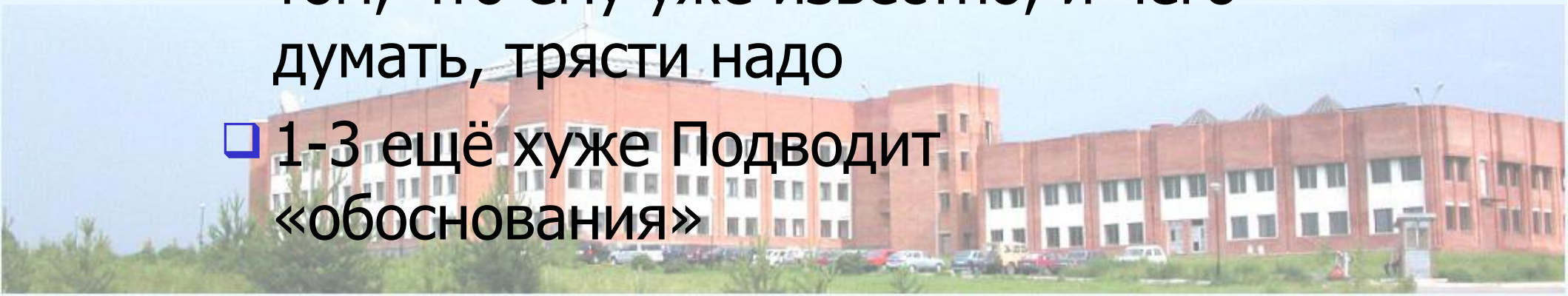
Уровни: примеры

- Хороший юрист – уровень знаний 1+ (виртуозно модифицирует тексты норм и предложений), уровень умений 0 (чтобы реальность не мешала воспринимать юридический мир)
- Хороший аналитик: уровень знаний 2 (легко модифицирует первоуровневые знания и соединяет их), уровень умений 1 (понимает связь с практикой)



Уровни: примеры -

- Математический идиотизм: математика 2-3 практика 0
- Ещё хуже: 3-1 С умным видом даёт никуда не годные советы
- Агрессивное невежесмтво программиста: практика 2 теория 0
- Думает, что всё можно сделать на том, что ему уже известно, и чего думать, трясти надо
- 1-3 ещё хуже Подводит «обоснования»



Практический вывод

- Для каждого вида деятельности нужна комбинация специалистов разного уровня.
- Желательный уровень большинства различается для разных работ.

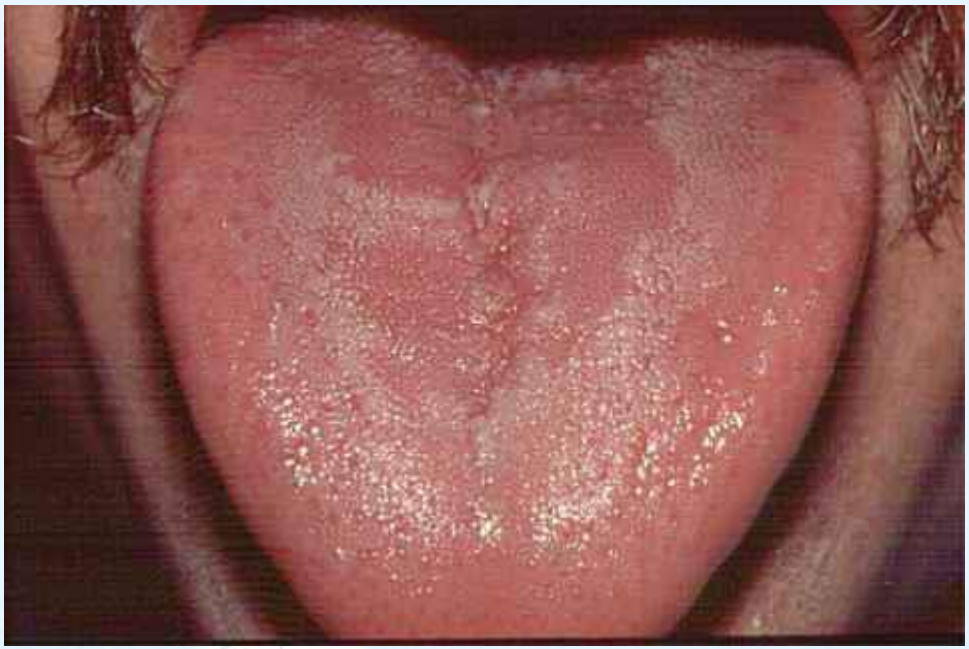
Диаграмма Злобина+

Диаграмма Злобина+

- На предыдущей диаграмме показана тонкими линиями субъективная уверенность хорошего специалиста в ценности информации. Важная с его точки зрения далее зашкаливает, а не теряет ценность.
- Средний вообще меняет местами знание и частную, а про важную обычно не имеет представления.

Почему?

- Программист изучал язык. Он с самого начала зомбирован думать в последовательных процессах и битах



Что необходимо 2

- Для суперкомпьютерной области необходимо для большинства уровень 3-2 в знаниях и умениях или наоборот.
- Программист (2-3) должен быстро перейти с одной системы на другую, поскольку системы привязаны к архитектуре и задачам.
- Он должен быстро понять изменение логики при изменении вычислительного базиса

Что необходимо 2

- Аналитик (3-2) должен осознать задачу в контексте не только алгоритма, но и его взаимосвязи с ресурсами и подсказать программисту нужное преобразование программы.
- При переходе от одного базиса к другому аналитик должен показать программисту, почему старые навыки вредны и какие новые умения нужны.

Обучение. Программирование

- Нужно учить не языкам, а всей системе стилей программирования.
- Нужно ко 2 курсу доводить студентов до уровня, когда преподаватель может сказать нечто типа: «В этих задачах скрипты пишутся на языке php. Вы за неделю его освоите сами, поэтому перейдём прямо к делу».

Обучение. Математика

- Высокоуровневому мышлению учит прежде всего математика.
- Для уровней 2-3, 3-2 вредна численная математика и физика, находящаяся на уровнях 1-2, 2-1
- Необходимо поменять местами приоритетность анализа (в широком смысле) и алгебры с логикой.
- Естествознание или общая культура: Химия. Биология. Астрономия. Лингвистика. Там структуры.

Обучение. Стиль

- Теория должна подаваться обязательно вместе с потенциальными применениями.
- Практика должна подаваться вместе с теоретическим осмыслением.
- Каждое привлекательное решение должно сопровождаться беспощадным показом недостатков, ограничений и опасностей.

Обучение. Стиль 2

- Необходимо, чтобы неразрывная связь достоинств и недостатков в любом человеческом решении вошла в подкорку.
- Никакой «актуальности» в фундаментальных курсах! Фундаментальное знание не устаревает за всю жизнь.
- Обязательный показ незаслуженно забытых старых решений(и заслуженно, но идейно интересных).

Обучение. Возможность

- Возможность довести 50% до 2-3, 3-2 при таком подходе проверена 20-летним опытом УдГУ.
- Но она базировалась на фундаменте подготовки советской школы: приходили ребята и девушки, чуть-чуть умеющие связывать знания и уже обладающие начальными навыками в информатике, а не в использовании гаджетов.

Обучение. Обоснование 1

Параллельные алгоритмы строятся по другим законам, чем последовательные. Для их создания нужно логическое мышление. <<Математическая логика>>, в которой полностью опущен раздел, посвящённый взаимосвязям формального и содержательного, лишь ещё более отупляет. Философская логика не лучше. Давать науку в комплексе .

Обучение. Обоснование 2

Созданная программа должна быстро преобразовываться. Навыки работы с высокоуровневыми структурами и преобразований систем даёт абстрактная алгебра, которая должна преподаваться как предельно конкретная (полугруппы=автоматы, группоиды= функциональные программы и т. п.)

И вновь к практике

Логически созданная программа с самого начала максимально совместна (из Алгола-68), что означает возможность легко уложить на любую подходящую для неё архитектуру.

Это даёт не оптимальное для конкретной аппаратуры решение, но на порядок лучше того, которое получается <<распараллеливанием>> последовательной программы.

И вновь к практике 2

Далее уже нужно одновременно и в комплексе видеть как алгоритм, так и ресурсные требования, что требует более высокого уровня знаний и умений.

Разные ресурсы — разные, несовместимые конструктивные логики.

Именно они лежат в основе построений

Нынешнее состояние

Кто, преодолев недостатки образования, вошёл в одну из архитектур (например, в матрично-конвейерную), теряется, когда надо программировать, скажем, на ассоциативном процессоре обработки данных.

Нынешнее состояние

Нужен опыт преобразования математических структур (алгебраические концепции).

Ситуация ещё обостряется в связи с перспективой перехода к алгебраическим процессорам.

Нынешнее состояние

Поэтому современный программист суперкомпьютеров, не знающий алгебры, может освоить максимум одну структуру и перестроить мозги под неё. При переходе на другую его ум полностью сломается. А в бурно развивающейся отрасли такое недопустимо.

Каковы общие следствия

- Если некоторое умение доведено до стадии навыка без того, чтобы снабдить вначале личность знаниями о его границах и недостатках, оно абсолютизируется и мешает воспринимать всё остальное.
- Умение, которое должно быть в дальнейшем отброшено, нельзя доводить до стадии навыка.
- Лозунг «Знания, умения, навыки» порочен в самой основе.