Контрольные вопросы:

- ☐ (5 б.) Почему класс `std::string` имеет много перегруженных функций-членов?
- ☐ (5 б.) Как осуществляется интернационализация и локализация программы?
- ☐ (5 б.) Чем отличаются многобайтовые кодировки от широких кодировок?
- ☐ (5 б.) Какие компоненты входят в стандарт кодирования символов Unicode?
- ☐ (5 б.) Для решения каких задач удобно использовать регулярные выражения?

Упражнения:

- ☐ (25 б.) Реализуйте систему конвертации валют, обеспечивающую ввод и вывод денежных сумм с учетом национальных стандартов. Например, Вы можете реализовать перевод EUR → RUB. В таком случае Вам потребуется использовать европейскую (например, немецкую) локаль для настройки ввода суммы в EUR и русскую (системную) локаль для настройки вывода суммы в RUB. Для наиболее удобной реализации предлагаю воспользоваться манипуляторами из стандартной библиотеки `std::get_money` и `std::put_money`.
- ☐ (25 б.) Реализуйте систему побуквенной транслитерации кириллических символов русского алфавита. Предположим, что пользователь вводит символы в консоли в кодировке системной локали по умолчанию, например, cp1251. Вы должны конвертировать кодировку символов из cp1251 в UTF-8, а затем в UTF-32 наиболее удобным способом. Напоминаю, для программной обработки текста предпочтительны кодировки, которые используют для представления символов постоянное число байтов. Далее замените все кириллические символы латинскими в соответствии с правилами транслитерации. Для программной реализации используйте заранее подготовленное отображение в виде хэш-таблицы. Конвертируйте кодировку обратно в UTF-8, а затем в cp1251 и выведите результат преобразования в стандартный поток вывода в консоль.
- ☐ (25 б.) Реализуйте систему, которая при помощи механизма регулярных выражений сможет находить в тексте все валидные email-адреса и выводить соответствующие им имена доменов. Найдите в интернете правила создания email-адресов и сформируйте на их основе регулярное выражение. Имя домена – это то, что стоит справа от символа @ в адресе. Проще всего выделять имя домена в адресе через использованние групп и подобразцов, т.е. имя домена должно будет попасть в одну из ячеек контейнера `std::smatch`. Для определения всех адресов в тексте используйте итераторы регулярных выражений. Текст подготовьте самостоятельно, т.ч. в нем присутствовали валидные и невалидные адреса и посторонний текст-мусор.
- ☐ (25 б.) Аналогично предыдущей задаче реализуйте систему, которая при помощи механизма регулярных выражений сможет находить в тексте все упоминания о датах и времени. Поскольку существует много форматов времени и даты, можете выбрать любой, например YYYY.MM.DD и HH:MM:SS – с лидирующими нулями. Обратите внимание, на некоторых позициях в определенных ситуациях могут стоять не все цифры.