**Rust Programming Lab #6 11th October 2022**

**What language is that?**

**Exercise 1**

In a global world, we are presented with documents, web-sites, text, etc in many languages now. However the Unicode standard makes it possible to, with reasonable accuracy, determine the source language. In this exercise, you should download a test file, work out the characters in it, determine which language they belong to, an make a report **on each line**, telling us what languages were used to write it and report the number of characters in that language.

**Steps**

1. Take the supplied text, read it (using **stdin** is recommended), determine the characters in it and determine the language used for each character. You can do this by ‘reverse lookup’, i.e. ask Google (or any other source) which language this Unicode belongs to. All languages are use blocks of Unicode characters, e.g. ASCII characters use Unicodes 0-127, Thai uses Unicodes 0E00 - 0E5B, etc.
	1. You should build an enumerated type containing readable labels for each language.
	2. Create a **struct** containing the enumerated type, the range of unicodes used by that language and a string used to print the name of that language.
	3. Build a table containing entries for these structs.

**Check point A(c)**

* 1. Build a function which, given a unicode value, scans the table and finds out to which language that character belong - returning the enumerated type.

**Check point A(d)**

* 1. You will need another function which, given a language, returns a string describing it.

**Check point A(e)**

1. Since looking up the unicodes individually may take several passes, an iterative approach is recommended.
2. Make an initial table which contains some expected unicodes, *e.g.* ASCII and Thai.

**Check point B(a)**

1. Read the test file, scan the table to find initially known languages (skip them) and list unicodes for languages not yet known.
2. Use the output from this program to look up remaining unicodes and add more entries to your table.

**Check point B(c)**

1. Repeat until all characters in the test file are assigned to a language.
2. Now write a program which reads the test file and produces a report, showing, **for each line**, which languages were used in that line and how many characters from that language were found. (Work out which type to use to create this report!)
So your report might look like:

Line 1: Thai (10), ASCII (4)
Line 2: Chinese (11), Japanese (4), ASCII (10), …..
Line 3: ….

Notes: Most languages have some ‘extra’ characters, e.g. ‘&’, ‘+’, ‘อ้’,’อั’, etc, For this exercise, if a unicode lies within the range for that language, *count it as belonging to that language*. ***Keep it simple!!***

**Answer the questions on the attendance sheet, have a TA sign it off. Prepare a short report and submit on goEdu. Do not forget to submit your code!!**

[ELEMENTARY SYSTEMS PROGRAMMING [Section 2][65-1-01286120-2]](https://goedu.kmitl.ac.th/course/view.php?id=10151)

What language

**before the next lab (Your deadline is 3pm, Oct 18.**

**For victims of Rust: Note that check points have been added to the steps listed. If you have any doubts, it is STRONGLY RECOMMENDED that you check with one of the TAs each checkpoint BEFORE proceeding to the next. At that point, you should also tell the TA what you are going to do next.**

**IF you are confident: You may skip the checkpoints and fill them in at the end.**

**Website: kris.kmitl.ac.th/clinic/Courses/Rust/**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attendance**  | **01286120**  | **Elementary Systems Programming** | **11 Oct 2022** |

|  |  |  |
| --- | --- | --- |
| **Name (Thai script\*)** |  | **Student ID** |
| **(Latin characters - as you enrolled)** |  |
| **\****Please write clearly: practice for one farang who is trying to improve* **😉** |

|  |  |  |
| --- | --- | --- |
| **CP A(c)** | Basic struct | TA |
| **CP A(d)** | Data type to hold list of structs |
| **CP A(e)** | Function find whether Unicode exists already? |
| **CP B(a)** |  |
| **CP B(c)** |  |
|  |
| **Check 2** | List languages and unicode ranges that you found | TA |
| **Language** | **Unicode range** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |