Pontificia Universidad JAVERIANA Cali | Facultad de Ingeniería y Ciencias

Proyecto 2 – Programación orientada a objetos

Noviembre 2021

Puede asumir información si necesita algo que no está explícito en este documento

ParkingSoft is a Colombian company that produces software solutions related to parking services. In recent years, this company has increased its market share to market segments in LATAM, United States and Canada and since June 2021 the company is planning to enter the Asian markets.

ParkingSoft is a multicultural organization as its founders are convinced that a multicultural environment has many advantages. For example, they believe that cultural diversity boosts innovation and creativity, increases respect, openness, and curiosity. Also, the founders are convinced that interculturality improves their employees' ability to tolerate ambiguity, which is a key attitude to compete in the global market.

All staff work in their home office and are located remotely around the world. However, sometimes they hold face-to-face meetings to networking and to strengthen personal relationships.

Considering that keeping an intercultural perspective is so important for the organization, people from the recruitment staff wants to improve their knowledge, skills and attitudes towards different cultures. For example, the recruiters want to better understand the candidate's behavior according to his/her culture to be respectful of his or her values and better prepare the interview process.

In order to help people from human talent, your team was delegated to design and elaborate on a software that will assist the recruitment team by providing information taking into account the diversity of cultures. A well-accepted definition of culture is the following:

> *"the sum of a way of life, including expected behavior, beliefs, values, language and living practices shared by members of a society. It consists of both explicit and implicit rules through which experience is interpreted". Hofstede, G. (2001) Culture's Consequences: International Differences in Work-Related Values, London: Sage*

This software would assist people from human talent with information and support to recruit people from different nationalities. The goal is to provide to the staff with a knowledge-base to better consider the culture factor when recruiting and on-boarding new employees.

Your software should satisfy the following requirement:

R1- Consult the cultural knowledge by nationality: Given any of the selected countries the software should provide information of the behavior for this specific nationality regarding, but not limited, to the following categories:

- Type: expressive culture | reserved culture
- Eye-contact style: duration and intensity of eye contacts
- Gestures; yes/no, Ok, good luck
- Touch: shaking hands, kissing, hugs, and other physical contact
- Respect for distance and physical space.
- Time management.
- Communication styles. [Volume, rhythm, tone]

Siesta traditions. For example in some hot countries there is a traditions of take a nap in the early afternoon.

R2 - Create the candidates in the system. The recruiter should be able to create in the system the candidates. Every candidate has a name, an email address, a LinkedIn URL, a Github URL and a passport number. Moreover, according to his/her nationality, the candidate will inherit the corresponding cultural information from the information defined in the system. Use the factory pattern to manage the creation of applicants.

R3 - Schedule interviews: to schedule an interview, the interviewer will give to the system the identification of the candidate. If the candidate exists, the system will define the date for the interview. Interviews start at 10 am. The system will assign 60 minutes for each candidate and a maximum of three interviews by day. If the candidate does not exist the system requests the interviewer to create the candidate in the system first.

R4- Generate interview guide. For candidates scheduled for an interview, the system must generate a text file including for each candidate the following information:

- Candidate's name and date of the interview and nationality.
- Information on national holidays according to the nationality of the candidate. This information will help the interviewer to ask icebreaking questions in the interviews.
- Expected intercultural behavior. String that explains the knowledge characteristics specific to the candidate's nationality. [Considering the categories described in RQ1]. You should design the software considering polymorphism to meet this functionality.

- **R5- Generate the welcome letter**. People from human resources could generate a letter for those candidates who passed the selection. This letter should inform the new employees of the cultural knowledge that make up Colombian culture (taking into account the same categories described in R1) and the values of the company. Below in this document you would find the a example of these values but your team would propose its own version [https://about.gitlab.com/handbook/legal/gitlab-code-of-business-conduct-and-ethics/]

Non functional requirements

- Use of the factory pattern.
- Use of polymorphism
- Elaboration of unit tests of the main functionalities.
- Exception handling for exception flows
- Besides Colombia your team should consider information from three different countries (at least)

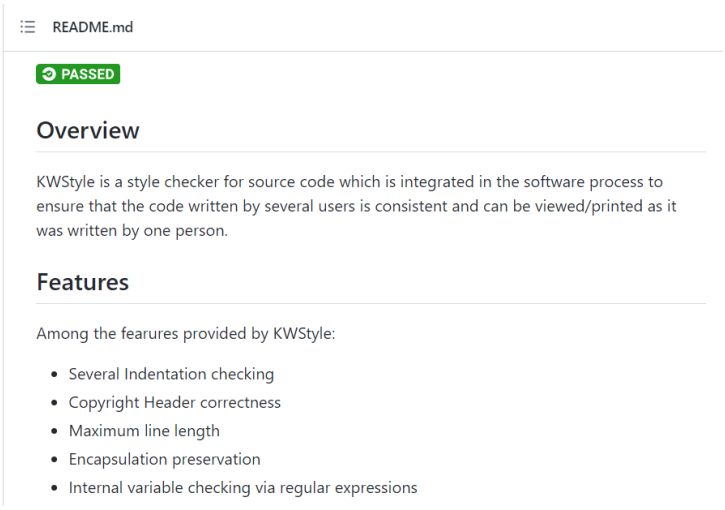
Mínimos esperados (no cuentan en la nota, pero sin esto no califico el proyecto):

En la elaboración de su programa debe considerar:

- Uso continuo de git para mantener el histórico de avance de su proyecto con comentarios claros sobre los cambios de su programa.
- Tener un menú usando do while y switch case para acceder a las diferentes opciones del programa.
- Cumplir con el estándar de codificación lowerCamelCase
- Código documentado para hacerlo claro para cualquier lector. En la documentación recuerde que sus comentarios deben explicar **por qué se hacen las cosas más que solo describir** literamente las líneas de código.
- Buenas prácticas de programación: buen nombramiento, no números mágicos, uso de constantes.

Entregables

- **Código fuente del programa.**
- **Informe de autoevaluación:** documento en el que explique qué problemas tuvo al hacer su proyecto, qué aprendió, qué le gustó, que no le gustó, qué hizo cada uno de los miembros del equipo, qué nota se pondrían de manera individual y qué nota le pondrían al compañero junto con la justificación. Cada persona del equipo debe entregar un informe individual.
- **README en el repositorio [Manual técnico]:** Con presentación general del proyecto, principales funcionales, y explicación de las principales funcionalidades de la aplicación funcionando y el diagrama UML (uno por equipo). La siguiente figura muestra una imagen de ejemplo de un README disponible en https://github.com/Kitware/KWStyle. Tenga en cuenta que para documentar el README en Github debe usar MarkDown (aquí puede encontrar más info https://www.markdownguide.org).


README.md

PASSED

Overview

KWStyle is a style checker for source code which is integrated in the software process to ensure that the code written by several users is consistent and can be viewed/printed as it was written by one person.

Features

Among the fearures provided by KWStyle:

- Several Indentation checking
- Copyright Header correctness
- Maximum line length
- Encapsulation preservation
- Internal variable checking via regular expressions

Todos los entregables deben ser subidos a un repositorio de git en una carpeta llamada ***Recruitment+iniciales***. En Teams uno de los miembros del equipo sube la URL de repositorio donde quedaron subidos los archivos. Puede subir la información a su repositorio máximo hasta las 11:55 pm del 28 de noviembre del 2021. Si prefiere dar la nota de su compañero de manera anónima puede comunicarse conmigo. ***Los documentos deben tener una calidad de redacción , ortografía y presentación esperadas a nivel universitario***. Las sustentaciones serán la siguiente semana

El lunes 22 y miercoles 24 estaré disponible para solucionar dudas asociadas al proyecto. También puede contactarme en otros momentos para solucionar estas dudas.

Calificación

- Autoevaluación: 10 %
- Calificación del compañero: 10%. En trabajos individuales la autoevaluación tendrá un 20% de peso. Equipos de máximo tres personas.
- Calificación de la profesora compuesta por entregables, diseño, funcionalidad, estilo de codificación y mejores prácticas: 80%. La rubríca de evaluación explica los elementos que se consideran para la calificación del proyecto.

- Propiedad intelectual: valor entre 0 y 1 que multiplica la calificación total. Se demuestra durante la sustentación.
- Nota final = (criterios evaluacion%) * propiedadIntelectual

Rúbrica de evaluación

La nota de la profesora será dada al finalizar la sustentación según los criterios que se describen a continuación y su capacidad para sustentar su trabajo.

	5	4	3	2	1	0
Entregables (10%)	Los informes contienen toda la información solicitada y tiene alta calidad en cuanto a estilo y formato.	Se entregaron todos los informes. En términos de contenido están completos pero podría mejorar en cuanto a estilo o formato	Se entregaron todos los informes. En términos de contenido están completos pero podría mejorar en cuanto a estilo Y formato	Faltan algunos de los informes pero los entregados tiene buen estilo y formato	Faltan algunos de los informes y los entregados necesitan mejoras de estilo y formato	No se entregaron los informes
Diseño (30%)	El diseño responde a los requisitos. Se detectaron todas las clases importantes y para cada clase se detectaron los atributos y métodos importantes. Usa las relaciones correctas	El diseño responde a los requisitos. Se detectaron todas las clases importantes y para cada clase se detectaron los atributos y métodos importantes. Tiene algunas relaciones incorrectas o faltantes en el diseño de los métodos como por ejemplo los atributos.	El diseño responde a los requisitos. Faltó detectar algunas clases importantes. Faltó detectar algunos de los atributos y métodos importantes. Tiene algunas relaciones incorrectas. El diseño no corresponde a lo codificado.	Faltó detectar la mayoría de clases importantes Faltó detectar muchos de los atributos y métodos importantes. Tiene muchas relaciones incorrectas	El diseño no satisface los requisitos	No se entregó el diseño
Funcionalidad (30%)	Cumplió con todos los requisitos.	Fueron desarrollados mínimo el 75% de los requisitos	El diseño responde al 75% de los requisitos / podría mejorarse	Fueron desarrollados mínimo el 25% de los requisitos	Fueron desarrollados menos del 25% de los requisitos	La funcionalidad no responde a lo solicitado
Estilo de codificación (5%)	El código se encuentra correctamente indentado, los nombres de los atributos y las funciones cumplen con el estándar de nombramiento. El código tiene documentación interna para facilitar la revisión.	La mayoría del código se encuentra correctamente indentado- La mayoría de los nombres de los atributos y las funciones cumplen con el estándar de nombramiento. La mayoría del código tiene documentación interna para facilitar la revisión	Falta alguno de los ítems de calidad del estilo de codificación o alguna se cumple con mala calidad	Faltan dos de los ítems de calidad del estilo de codificación o dos se cumple con mala calidad	No hay código fuente suficiente para evaluar el estilo de codificación.	No cumple con el estándar de nombramiento No se encuentra correctamente indentado No está divido adecuadamente

Mejores prácticas (5%)	El código muestra mejores prácticas de desarrollo siempre. Reúso, separación de operaciones, buen manejo de ciclos, simplicidad, programación defensiva validaciones.	El código muestra mejores prácticas de desarrollo en la mayoría de los casos, pero falta mejorar algunos de los siguientes aspectos Reúso, separación de operaciones, buen manejo de ciclos, simplicidad, programación defensiva validaciones	El código muestra buenas prácticas de desarrollo pero falta mejorar dos de los siguientes aspectos Reúso, separación de operaciones, buen manejo de ciclos, simplicidad, programación defensiva validaciones	El código aplica pocas buenas prácticas de desarrollo. Falta mejorar tres de los siguientes aspectos: Reúso, separación de operaciones, buen manejo de ciclos, simplicidad, programación defensiva validaciones.	No hay código fuente suficiente para evaluar las buenas prácticas.	No se entregó proyecto o el proyecto no cubre al menos el 25% de las funcionalidades. No es posible evaluarlo .

Rúbrica de propiedad intelectual

	1	0.8	0.6	0.4	0.2	0
Sustentación	Es evidente que el estudiante entiende el código que desarrolló lo explica con claridad y responde correctamente a las preguntas.	La sustentación es buena pero se evidenció inseguridad del estudiante para explicar algunas partes del trabajo desarrollado o para responder algunas preguntas.	La sustentación es aceptable se evidencia que el estudiante desarrolló el código pero le cuesta trabajo explicar aspectos del código.	La sustentación es regular se evidenció inseguridad del estudiante para explicar gran parte del trabajo desarrollado o para responder muchas de las preguntas. Parece que el código no hubiera sido desarrollado por el estudiante.	El estudiante demuestra que entiende partes del código, pero no tiene claro cómo se relacionan con la funcionalidad solicitada.	Se evidencia que el estudiante no entiende el código desarrollado, no es capaz de responder a las preguntas formuladas de manera correcta.