**FORMATO PARA EL DESARROLLO DE COMPONENTE FORMATIVO**

|  |  |
| --- | --- |
| PROGRAMA DE FORMACIÓN | Desarrollo de aplicaciones web con Python |

|  |  |  |  |
| --- | --- | --- | --- |
| COMPETENCIA | 220501123 - Construir sitios web según técnicas de interoperabilidad y protocolos técnicos. | RESULTADOS DE APRENDIZAJE | 220501123-01 - Elaborar algoritmos con Python de acuerdo a las necesidades del cliente. |

|  |  |
| --- | --- |
| NÚMERO DEL COMPONENTE FORMATIVO | 001 |
| NOMBRE DEL COMPONENTE FORMATIVO | Entornos de desarrollo y codificación |
| BREVE DESCRIPCIÓN | Este componente formativo está orientado a entregar los conceptos y conocimientos en instalación y puesta en marcha de un entorno de trabajo. También en el dar a conocer los estándares de codificación, manejo de variables, funciones, estructuras de control, clases objetos y gestión de proyectos en Python. |
| PALABRAS CLAVE | Estándar, variables, funciones, clases, objetos. |

|  |  |
| --- | --- |
| ÁREA OCUPACIONAL | VENTAS Y SERVICIOS |
| IDIOMA | Español |

1. **TABLA DE CONTENIDOS:**

**1. ¿Qué es un editor de código?**

**2. Diferentes editores de código**

**3. Proceso de instalación de *visual estudio code***

**4. Preparar entorno de desarrollo virtual**

**5. Manejo y control de versiones**

**6. Estándares para la codificación de *software***

**7. Creación y manejo de variables, estructuras de control, funciones, clases y objetos**

**8. Gestión de proyectos en Python**

1. **INTRODUCCIÓN**

Bienvenidos a este componente formativo en el cual se abordarán temáticas de suma importancia para el desarrollo de nuestro programa, sus contenidos están orientados a sumergirnos en un mar de conocimiento y aprendizaje sobre los conceptos y conocimientos de instalación y puesta en marcha de un entorno de trabajo en el lenguaje de programación Python. Se invita a ver el siguiente video:



1. **DESARROLLO DE CONTENIDOS:**
2. **¿Qué es un editor de código?**

Un editor de código es una herramienta que permite realizar el proceso de creación de un archivo de código fuente donde se colocan todas las instrucciones necesarias para que un programa pueda ser ejecutado de manera correcta.

Los editores de código actualmente permiten la integración de diversos lenguajes de programación, lo cual facilita tanto la escritura como la modificación de código fuente. Adicionalmente, los editores de código por su poco uso de recursos de computadora son una gran alternativa para aquellas máquinas que no poseen una gran capacidad de cómputo para la ejecución de código fuente.

Existen muchos aspectos que se pueden rescatar de los editores de código, pero una de las particularidades más asombrosa que se puede tener en cuenta es la capacidad de adaptación a la estructura del código que se está escribiendo en el mismo. En pocas palabras, tiene la capacidad de adaptar el estilo del código de acuerdo con el lenguaje de programación que se esté utilizando.

Un ejemplo de ello es la sintaxis, por ejemplo, el lenguaje Python comparado con el lenguaje C# es totalmente distinto. En el lenguaje C# se puede utilizar llaves para abrir condicionales, ciclos y funciones, las cuales se pueden colocar en cualquier parte del código para darle un estilo de tabulación adecuado a ese diseño específico; es decir, adapta la tabulación del código de acuerdo con ese lenguaje. La sintaxis de Python, por su parte, no permite que se coloque una línea de código debajo de alguna instrucción; para evitar esto, el editor de código coloca un espacio de tabulación en el comando que se encuentra debajo de un comando *if*, *else, for, while* entre otros, con el fin de no generar un error por tabulación del código. Estos aspectos hacen de un editor de código una herramienta que ayuda a organizar el código fuente de manera sencilla y eficiente al programador y es muy valioso al momento de realizar la codificación de un programa.

En la siguiente figura se evidencia la manera en cómo un editor de código, en este caso *visual studio*, realiza el ordenamiento de la estructura del código fuente para evitar que puedan existir errores y también ayudar en el aspecto visual al usuario para identificar las líneas de código de una manera mucho más comprensible, contribuyendo, además, a la solución de errores o problemas que se puedan presentar en el momento de realizar la codificación del programa; este es un claro ejemplo de las ventajas que ofrecen los editores de código a los desarrolladores de *software*.

**Figura 1**

*Estructura de código fuente en Python*

Captura de pantalla de computadora

Descripción generada automáticamente

Nota. Pantalla en *visual studio code.*

1. **Diferentes editores de código**

En la actualidad existen muchos editores de código que se encargan de ayudar a los programadores de aplicaciones a realizar este proceso de manera organizada y estructurada. Se debe tener en cuenta que estas herramientas están con el propósito de mejorar la codificación de aplicaciones.

También, es importante conocer que muchas de estas herramientas deben ser seleccionadas por el programador, de acuerdo con las necesidades que se tengan para el desarrollo del proyecto a construir. A continuación, se presentarán algunas de ellas:



Todas estas herramientas están a disposición para cualquier persona que desee incursionar en el mundo de la programación, son de descarga gratuita y poseen licencia “*Free*” (es decir, se pueden usar sin tener que pagar). Estas herramientas también permiten la integración de la mayoría de lenguajes de programación, lo que facilita la depuración y codificación de aplicaciones informáticas en la actualidad; por lo tanto, son alternativas muy utilizadas en la comunidad de programadores a nivel mundial.

Sumado a lo anterior, estas herramientas incluyen actualizaciones continuas en las que se agregan nuevos componentes tales como *plugins.* Esto le permite a los desarrolladores integrar tecnologías y rutinas para mejorar la forma en cómo desarrollan las aplicaciones.

Una de estas herramientas que se traen a colación es *TABNINE.* Es un *plugin* utilizado en diversos editores de código que trabaja con tecnología *Deep Learning*, la cual puede predecir si se utilizará una rutina específica o código. Esto sucede al llevar un registro del estilo de programación de cada persona y luego la herramienta ayuda a reutilizar dichas rutinas y a construir el código fuente de una manera mucho más sencilla y organizada.

**3. Proceso de instalación de *visual studio code***

Como se ha abordado hasta este momento, existen diferentes editores de código, los cuales permiten realizar el proceso de codificación de las aplicaciones informáticas. Cada uno de ellos ofrece diversas ventajas y desventajas, llegando a un punto de no poder establecer cuál de todos es el mejor, sino que de forma individual cada programador elegirá aquel con el que se sienta más cómodo y terminará convirtiéndose en su editor de preferencia.

Sabiendo esto, se presentan los pasos a seguir para realizar la instalación de la herramienta en el computador; para este caso *visual studio code,* el editor que se utilizará para el desarrollo temático. Por favor tener en cuenta cada paso que se deberá llevar a cabo para realizar la instalación de esta herramienta de manera correcta:

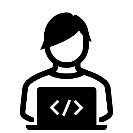
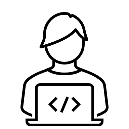


**4. Preparar entorno de desarrollo virtual**

A continuación, se ven los pasos a seguir para la configuración de un entorno virtual de desarrollo. Es importante seguirlos como se indica para realizar este proceso de manera correcta; para esto, se debe comenzar con una exploración de qué es un entorno de desarrollo virtual.

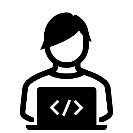
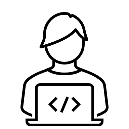
Un entorno de desarrollo virtuales la creación de directorios aislados, los cuales permiten realizar la configuración de los componentes que sólo requiere el proyecto sin necesidad de afectar todo el sistema en general.

Por ejemplo, al suponer que se tienen dos aplicaciones una llamada App1 y otra llamada App2.

App 1 App 2

Para el primer caso se requiere la instalación de las librerías L1 y L2 y para el segundo se necesita solo instalar la librería L2.

 Descargar contornoDescargar contorno

App 1 App 2

En este caso la instalación se realizaría de manera independiente para cada aplicación, sin que todas se vean afectadas, ya que cada una tiene su propio espacio de configuración y de instalación de componentes necesarios.

Esta es una excelente estrategia cuando se trabaja con sistemas de versionamiento, que en versiones superiores ya no se usen componentes y no se requiera instalar en las nuevas versiones.

Se pueden ver entonces los pasos para la instalación de un entorno de desarrollo virtual:



Como se ve a través de esta ejemplificación, la creación de entornos de desarrollo virtuales ofrece grandes ventajas al momento de realizar desarrollos orientados al uso de librerías y extensiones propias de un proyecto independiente; el tema de versionamiento también toma gran fuerza, ya que se pueden establecer todos los componentes que este llevará y es mucho más fácil instalar y colocar en marcha las aplicaciones y su despliegue en servidores de producción.

**5. Manejo y control de versiones**

Para el manejo y control de versiones que se realizan a los proyectos es importante tener en cuenta que existen muchas herramientas en el mercado que se pueden utilizar para llevar a cabo este proceso.

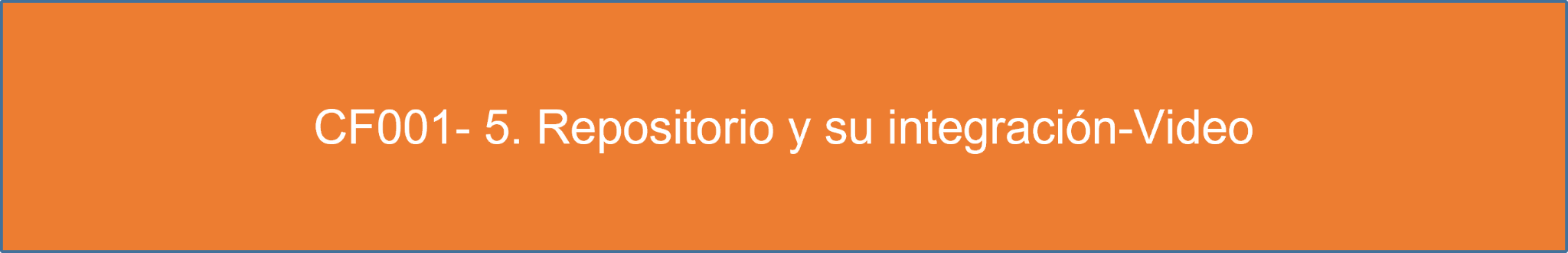
Para este caso en particular de este componente formativo, se ha establecido trabajar con dos herramientas que son líderes a nivel mundial (*Git* y *Visual Studio Code*) para realizar este proceso, por lo cual se abordará la manera en cómo estas son usadas y su importancia en los proyectos de software:

**Git:** es un programa que permite controlar las versiones que se trabajan en un *software*. Esto contribuye en el mejoramiento y desempeño de las aplicaciones, lo que permite realizar una actualización de los archivos de código fuente que se trabajan en el proyecto de *software*. También permite llevar un control detallado de todos los sucesos que ocurren al interior del código y así poder realizar cambios y que estos queden registrados para un mayor control. Esto ayuda a que se controle quién y cuándo se realiza un cambio en el código fuente de la aplicación.

Con Git es importante que se tenga en cuenta que esta herramienta tiene un entorno gráfico para realizar la administración de los procesos que se llevan a cabo con ella; entonces, lo primero que se hará es realizar su instalación. Se deja un video en el material complementario denominado **“Instalación de Git en Windows paso a paso”** para que se pueda observar el proceso de instalación de la herramienta Git.

**Nota:** en el video del material complementario se explica de manera detallada cómo realizar la instalación de la herramienta Git. Cuando se le pregunte el editor de código que se integrará, en ese paso se debe seleccionar la opción que dice “*Visual studio code*” para que el editor de código quede conectado con la herramienta Git. También se puede ver que existe una versión de Git llamada GUI, la cual permite crear los repositorios de una manera mucho más cómoda, ya que tiene una interfaz gráfica muy sencilla de operar.

A continuación, se explica cómo crear un repositorio utilizando la herramienta Git y luego cómo integrarlo a *visual studio code*; es importante estar atentos a los pasos para lograr realizar el proceso de manera correcta:



**Nota:** para mayor comprensión, es importante tener en cuenta que Git y GitHub son dos entidades diferentes que ayudan a administrar y alojar archivos. En otras palabras, Git sirve para controlar las versiones de los archivos gestionando el código fuente, mientras que GitHub es una plataforma para alojar o descargar repositorios Git.

Como conclusión, se ha trabajado con los repositorios de códigos, los cuales permiten llevar un control de los cambios realizados en el código fuente y adicional a ello tener una copia en la nube para darle mejor seguridad a los archivos de código fuente de la aplicación.

Es importante tener en cuenta que este tipo de aplicaciones son muy utilizadas en las empresas y equipos de desarrollo a nivel mundial ya que proponen un nivel de seguridad alto para el código, seguimiento y la colaboración.

**6. Estándares para la codificación de *software***

Conocer la importancia del proceso de codificación y su manejo de estándares es importante para abordar y comprender esta temática de una manera mucho más específica en cuanto a su uso y aplicación.

El concepto estándar es un modelo que se toma como referencia para construir, diseñar o fabricar un producto. Los estándares tienen en su teoría una serie de pasos y reglas que se deben cumplir para garantizar que la calidad y/o procesos se realicen de manera correcta.

Un estándar garantiza que un producto sea bueno o de calidad ya que se siguen ciertas reglas de fabricación y medición en cada fase del proceso.

Un ejemplo de lo anterior, puede ser la fabricación de una motocicleta:



Ahora bien, no solamente se hace referencia al proceso de fabricación, sino que también se debe tener en cuenta que se fabrica la motocicleta por primera vez y esta sirve de referencia para todas las demás fabricadas, ya que ha sido probada y ajustados sus valores de calibración y fondo que se tendrán en cuenta para todas las demás.

Así mismo, funciona en programación, el estándar sirve para mantener un estilo y una forma de escribir el código fuente y que este sea interpretado de una manera estandarizada.

Eso quiere decir que pueda ser entendible por otros programadores para cuando se requiera realizar una modificación o agregar nuevas funciones y se pueda realizar este proceso sin traumatismos causados por el no entendimiento. Esto incluye nombres de variables, nombre de funciones, atributos, clases, entre otros componentes.

Ahora bien, sabiendo lo que representan, es importante presentar los estándares de programación más utilizados para Python:

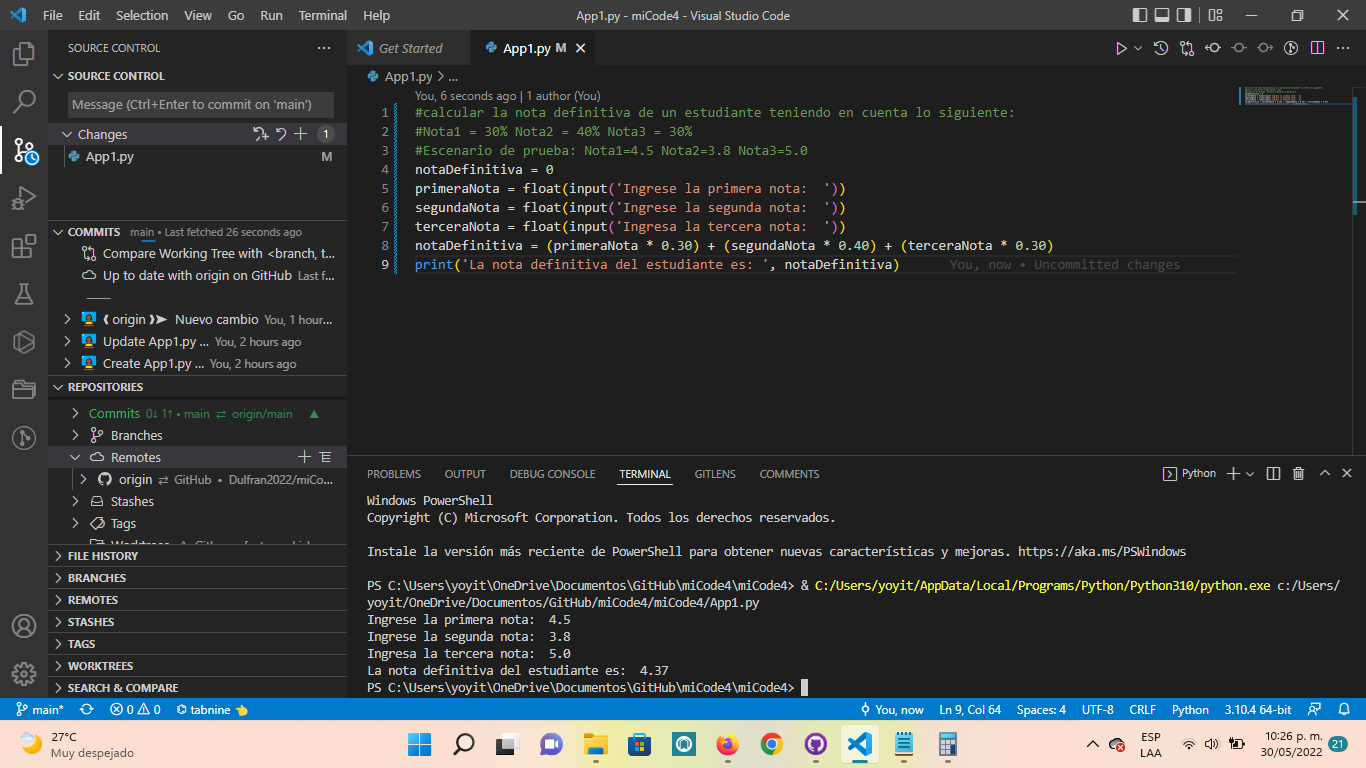


***Camel Case.*** Es el estándar más utilizado por la mayoría de las comunidades de desarrolladores en el mundo y se ha convertido en una de las que mayor referencia tiene en el código fuente que se ha estudiado. Es el preferido por los programadores por su sencilla manera de uso y practicidad.

Se puede ver en la siguiente figura 2 un ejemplo de *Camel Case* con Python:

**Figura 2**

*Estándar Camel Case en Python*



Nota. Pantallaen *visual studio code.*

Como se puede observar se utiliza el estándar de codificación *Camel Case*, uno de los más utilizados y preferido por los programadores como se comentó previamente.

El proceso de construcción del código sigue un patrón específico, pero es importante tener en cuenta que la manera en cómo se estructura el código es fácil de interpretar y de modificar las veces que sea necesarias, también se debe mencionar que este tipo de forma de codificar permite que otros programadores o miembros de un equipo de desarrollo puedan entender y trabajar de una manera mucho más fluida y ordenada.

Los estándares de desarrollo son una manera de no solo organizar el código, sino de trabajar en equipos de desarrollo.



Estos aportan calidad al producto de *software* que se desea construir y se debe tener en cuenta la palabra calidad, ya que el *software* es un producto y como tal requiere no solo de unos estándares para reflejar calidad, sino que también requiere llevar a cabo un proceso de documentación donde se consagra un documento llamado el manual del programador.

En este, se documenta el código y cada una de las funciones que realizan ciertas líneas de código y dicho documento se utiliza para generar una guía por si se requiere más adelante modificar o agregar nuevas funcionalidades al *software*, se tenga un punto de partida para que el proceso se lleve a cabo con los menores contratiempos esperados.

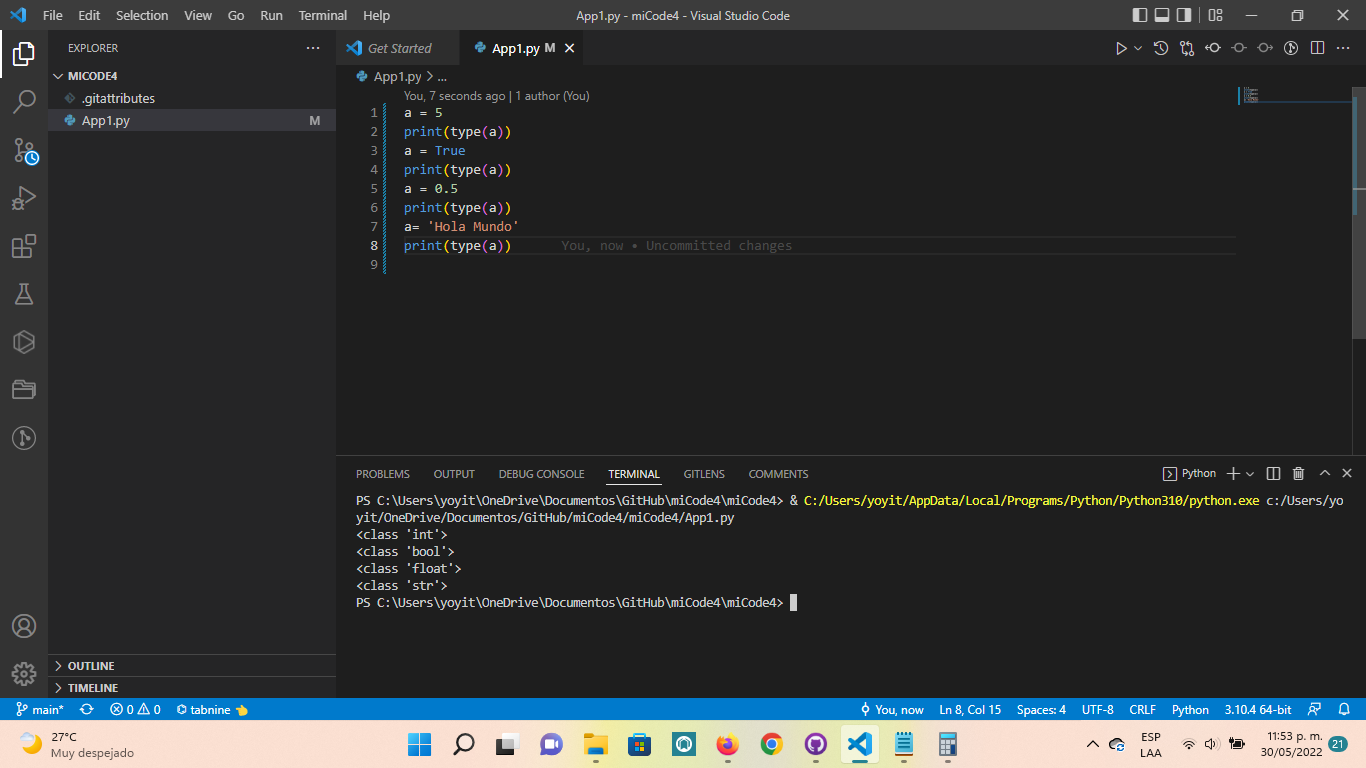
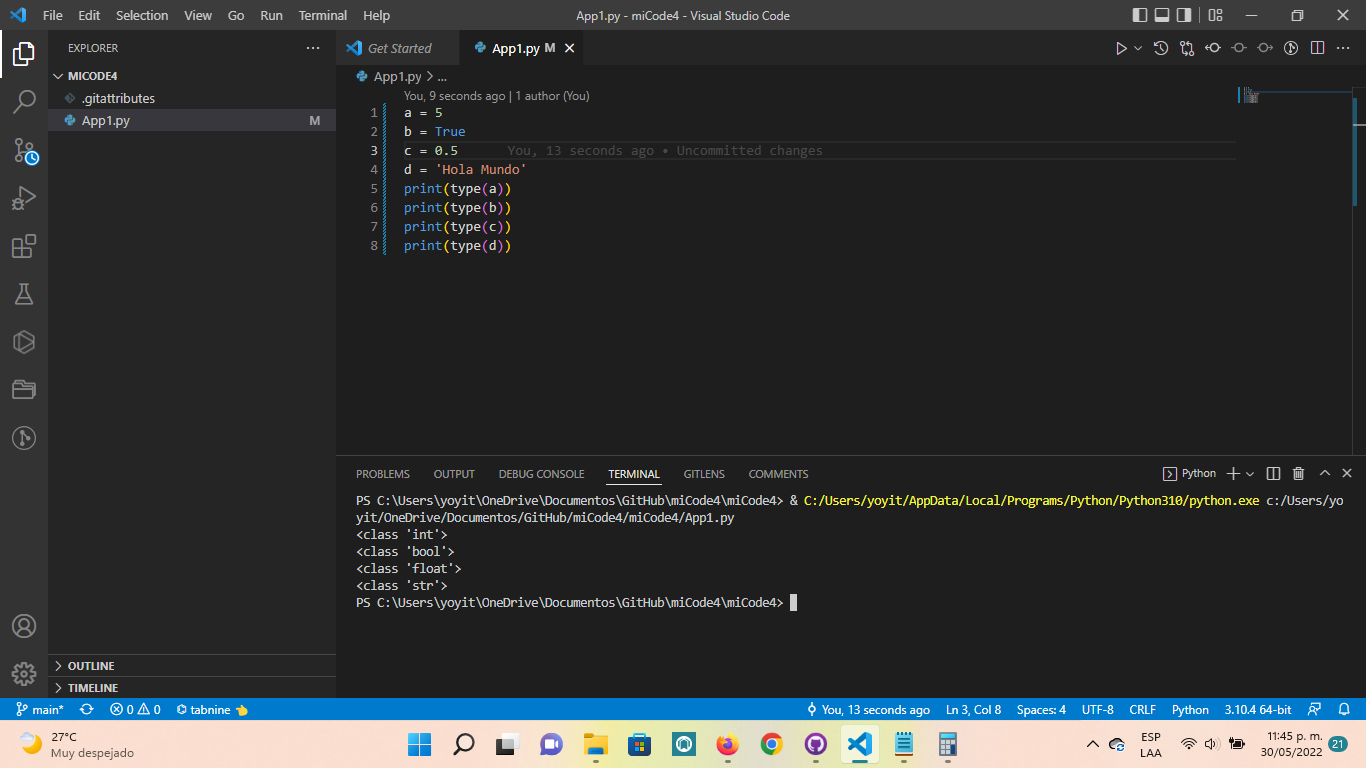
**7. Creación y manejo de variables, estructuras de control, funciones, clases y objetos**

En Python es fundamental el manejo de variables, teniendo en cuenta que este lenguaje de programación propone un uso muy particular de las mismas, permitiendo que una variable pueda recibir prácticamente cualquier tipo de información. Es decir, que en Python a diferencia de otros lenguajes de programación las variables, dependiendo del valor que se les asigne, así es su tipo de dato.

A continuación se ve un ejemplo de ello (ver figura 3):

**Figura 3**

*Ejemplos de sintaxis de las variables*



Nota. Pantalla en *visual studio code.*

Como se puede observar en la figura las variables (a, b, c y d) tienen asignados diferentes tipos de datos y con la utilización de la función *type* permite saber qué tipo de variable es. En otros lenguajes de programación es necesario declarar el tipo de dato que soportará la variable, pero en el caso de Python una misma variable puede cambiar de tipo de dato sin tener ningún inconveniente.

Esto refleja lo poderoso y versátil que es el lenguaje Python, en este caso particular se ha utilizado la misma variable para almacenar diferentes tipos de datos. Esta particularidad genera una ventaja al momento de realizar la codificación de un software puesto que se pueden reutilizar variables y por supuesto hacer un uso mucho más eficiente de la memoria RAM del equipo.

A continuación, se pueden ver los conceptos que junto al manejo de variables forman parte del entorno de trabajo Python. Ahora es tiempo de ver las estructuras de control, funciones, clases y objetos y conceptos claves para desarrollar en lenguaje de programación Python.

**Estructuras de control**

Para comenzar, en los lenguajes de programación, las estructuras de control permiten modificar el flujo de ejecución de las instrucciones de un programa.

Las estructuras de control permiten, a través de que se cumpla alguna condición, realizar la ejecución de una secuencia de código.

En la programación son muy importantes, así como las funciones, las clases y objetos. Los programas de computadora deben decidir de acuerdo con la información que se le suministra al sistema, realizar validaciones de acuerdo con las necesidades que este tenga que ejecutar y luego entregar al usuario los resultados esperados.

A continuación se pueden ver todos estos conceptos relacionados:

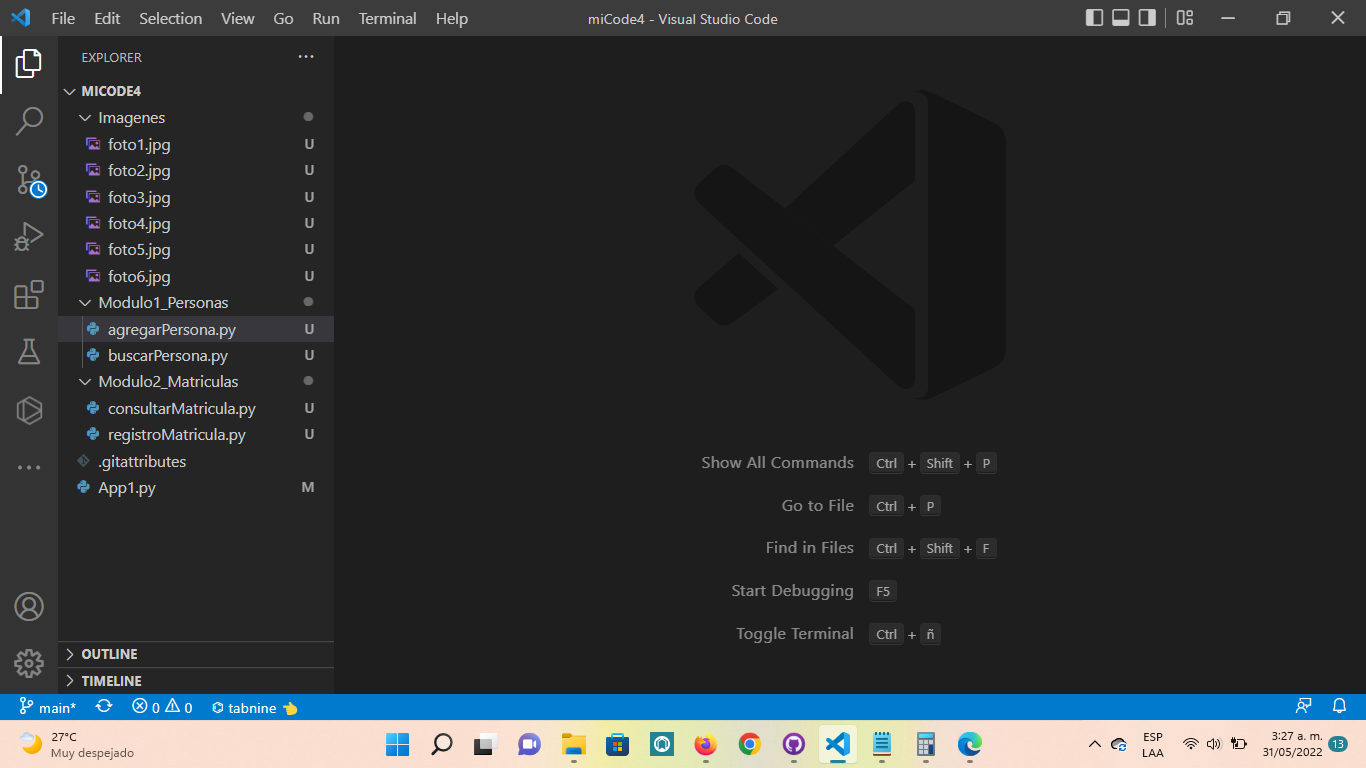


A continuación, se puede mostrar a través del siguiente video una ejemplificación de lo visto hasta este momento, en el que se irá también familiarizando con los conceptos y uso del aplicativo Visual Studio Code seleccionado:



**8. Gestión de proyectos en Python**





La gestión de proyectos en Python es la forma en cómo se pueden administrar diferentes espacios, carpetas y directorios que se puedan llegar a utilizar dentro del aplicativo a desarrollar.

Es importante tener en cuenta que esto va a permitir organizar de manera mucho más específica el programa, ya que de acuerdo con las necesidades que este requiera se crearán dichos espacios.

Como se puede ver en la imagen del proyecto en Python, se requiere el uso de más de una carpeta para su organización.

En la imagen se visualiza la manera en cómo se puede gestionar y visualizar un proyecto en Python, a diferencia de los observados anteriormente este se encuentra distribuido en diferentes carpetas y directorios los cuales de acuerdo con la necesidad del programador puede agregar o quitar carpetas y directorios.

Nota. Pantalla en visual studio code

Es importante tener en cuenta que la gestión del proyecto no solamente se requiere para la realización y organización del proyecto en sí, sino que también permite realizar una interpretación de cómo está distribuido el programa en términos de lógica.

Un ejemplo de esto es que muchas veces en las empresas cuando los proyectos son demasiado extensos se procede a asignar responsabilidades funcionales a diferentes miembros del proyecto. Por ejemplo, a un programador se le puede asignar el módulo de factura y al otro el módulo de inventarios, ambos módulos pertenecen al mismo proyecto pero están en dos espacios distintos; esto ayuda adicionalmente a mejorar la seguridad y el orden dentro del proyecto.



Ahora bien, es importante comprender que el uso de las herramientas que se han estado explorando a través de este componente formativo juegan un papel importante, ya que permiten en este caso particular controlar ciertos aspectos del proyecto que requieren de un gran cuidado y verificación constante.



Como es el caso de los repositorios para el almacenamiento del código fuente del programa, se hace referencia a la herramienta GitHub, esta herramienta permite almacenar en un servidor de repositorio el código fuente del programa de una manera segura y sencilla. En muchas ocasiones uno de los problemas que se presentan en el trabajo de proyectos de *software* es precisamente dónde reposarán los archivos fuentes ya que estos son los que no solamente tienen las instrucciones y la lógica para hacer funcionar el proyecto, sino que también van a permitir realizar la modificación o agregación de nuevas funcionalidades al programa.

En algunas ocasiones se almacenan estos archivos en discos duros y/o computadoras de respaldo las cuales pueden llegar a fallar en cualquier momento; por eso, la recomendación **siempre** está dirigida a la utilización de este tipo de servicios (como GitHub) para evitar pérdidas de archivos o en su defecto código fuente construido.

Se puede poner como ejemplo el siguiente escenario:

Texto

Descripción generada automáticamente con confianza media

Icono

Descripción generada automáticamente 





Imagen que contiene Forma

Descripción generada automáticamente

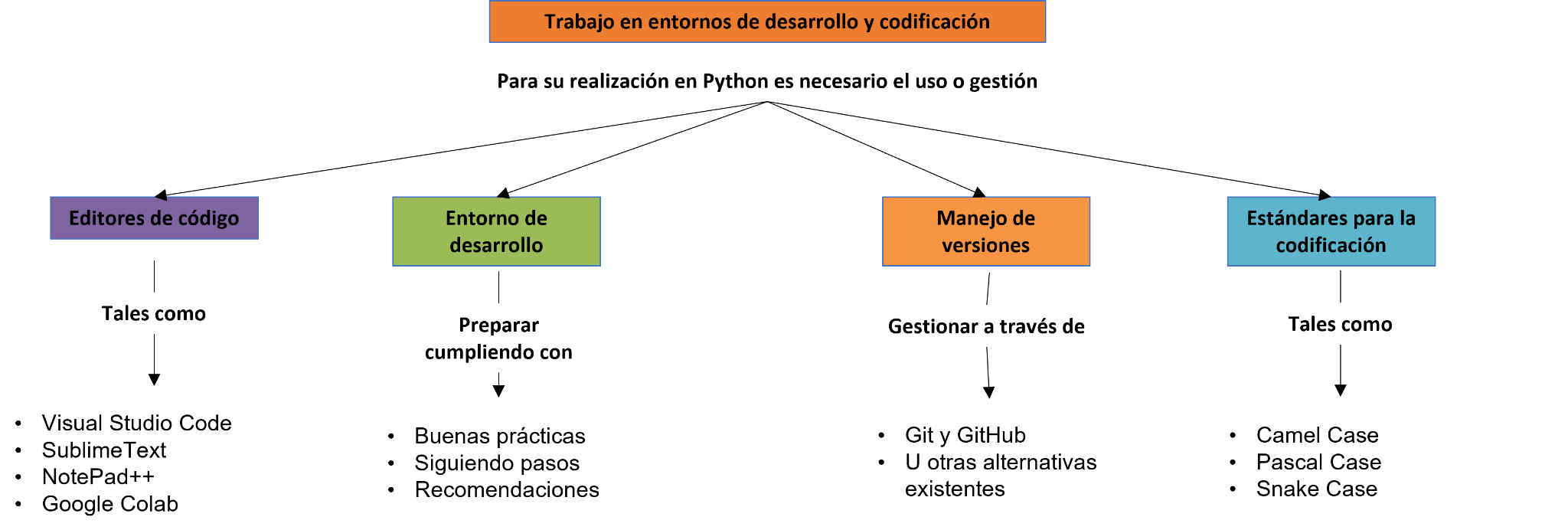
Todo lo desarrollado hasta este momento, lleva a la reflexión sobre la importancia de almacenar el código fuente de manera segura y confiable para evitar que este se pueda perder o pueda ser afectado por temas de virus o cualquier otro tipo de amenazas que se encuentran en el día a día de la industria del *software*.

También es importante la utilización de las buenas prácticas al momento de codificar, el uso de una aplicación para hacer que el programa mantenga un esquema de orden y nomenclaturas que permitan tanto su modificación como agregación de nuevas funciones de manera sencilla y eficiente.

Finalmente, se ha podido explorar las diferentes estructuras de control que se utilizan para validar las opciones y operaciones que se deben llevar a cabo dentro de un programa, el manejo de clases y objetos para organizar la información de una manera adecuada. Es importante tener en cuenta que el manejo de funciones permite separar las diferentes operaciones que se deben realizar en el programa y cómo este debe recibir la información que le entregará al usuario y la respuesta que se debe dar de acuerdo con los procesos necesarios por los requerimientos y necesidades del cliente para su desarrollo.

1. **SÍNTESIS**

Para garantizar la calidad del trabajo, la satisfacción de todos los públicos involucrados y cumplir con los requerimientos y necesidades del cliente, es necesario el cumplimiento de los pasos y puntos vistos a través del componente, para que, junto a una correcta Gestión del Proyecto, se logre un adecuado entorno de desarrollo y codificación; teniendo en cuenta lo siguiente:

****

1. **ACTIVIDAD DIDÁCTICA**

|  |  |
| --- | --- |
| DESCRIPCIÓN DE ACTIVIDAD DIDÁCTICA | |
| Nombre de la Actividad | Conociendo mi entorno de trabajo |
| Objetivo de la actividad | Validar los conceptos sobre los entornos de desarrollo y codificación y su aplicación en entornos de trabajo teniendo en cuenta el lenguaje de programación Python. |
| Tipo de actividad sugerida |  |
| Archivo de la actividad  (Anexo donde se describe la actividad propuesta) | Anexos / Actividad\_CF001 |

1. **MATERIAL COMPLEMENTARIO:**

|  |  |  |  |
| --- | --- | --- | --- |
| Tema | Referencia APA del Material | Tipo de material  (Video, capítulo de libro, artículo, otro) | Enlace del Recurso o  Archivo del documento o material |
| 1. ¿Qué es un editor de código? | Visual Studio Code. (s.f.). *Getting Started*. Docs. | Manual de referencia | <https://code.visualstudio.com/docs> |
| 5. Manejo y control de versiones | Dauzon, S. (2022). *Git: Controle la gestión de sus versiones (conceptos, utilización y casos prácticos).* Ediciones ENI. | Libro | <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1j5choe/sena_biblioteca_eniEPT3GIT> |
| 5. Manejo y control de versiones | Don Eber. (2021). *Instalación de Git en Windows paso a paso | [2021|2022]*. [Video] YouTube. | Video | <https://www.youtube.com/watch?v=cYLapo1FFmA> |
| 5. Manejo y control de versiones | Espitia, W. (2021). *Cómo crear una cuenta de GitHub desde cero en 2021.* [Video]. YouTube. | Video | <https://www.youtube.com/watch?v=jwFSIEi_d7E> |
| 7. Creación y manejo de variables, estructuras de control, funciones, clases y objetos | Commit That Line! (2020). *Las funciones en Python | ¿Para qué sirven y cómo se usan?* [Video]. YouTube. | Video | <https://www.youtube.com/watch?v=hLRoDs4wNCU> |
| 8. Gestión de proyectos en Python | Hinojosa Gutiérrez, Á. P. (2016). *Python: Paso a paso*. Ediciones de la U, Ra-Ma. | Libro | <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1j5choe/sena_ebooks0004998> |
| 8. Gestión de proyectos en Python | Python.org (s.f.). *Python 3.10.5 documentation.* | Manual de referencia | <https://docs.python.org/3/> |

1. **GLOSARIO:**

|  |  |
| --- | --- |
| TÉRMINO | SIGNIFICADO |
| Clase | Es una plantilla para el objetivo de la creación de objetos de datos según un modelo predefinido. Las clases se utilizan para representar entidades o conceptos, como los sustantivos en el lenguaje. |
| Editor de código | Es un programa que permite la creación y edición de archivos de código fuente. |
| Entornos virtuales de desarrollo | Es la creación de directorios aislados los cuales permiten realizar la configuración y componentes que sólo requiere tu proyecto sin necesidad de afectar todo el sistema en general. |
| Estructuras de control | Son utilizadas para validar y tomar decisiones dentro de un programa y que a su vez permiten la ejecución de bloques de código. |
| Función | Las funciones son como una especie de caja las cuales pueden contener una serie de códigos los cuales pueden ser invocados las veces que sea necesaria dentro del programa. |
| GitHub | es un repositorio de código que se encuentra alojado en la nube y se utiliza para guardar archivos de código fuente y realizarle seguimiento a los mismos con respecto a los cambios o actualizaciones que estos puedan recibir. |

1. **REFERENCIAS BIBLIOGRÁFICAS:**

Dauzon, S. (2022). *Git: controle la gestión de sus versiones (conceptos, utilización y casos prácticos).* 2ª Edición. ed. Ediciones ENI. <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1j5choe/sena_biblioteca_eniEPT3GIT>

Hinojosa Gutiérrez, Á. P. (2016). *Python: Paso a Paso*. Primera Edición. ed. Ediciones De La U, Ra-Ma, <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1j5choe/sena_ebooks0004998>

1. **CONTROL DEL DOCUMENTO**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Nombre | Cargo | Dependencia | Fecha |
| Autor (es) | Dulfran Antonio Montaño Montaño | Experto Temático | Centro De Diseño Y  Metrología - Regional  Distrito Capital | Julio de 2022 |
| Zvi Daniel Grosman Landáez | Diseñador Instruccional | Regional Distrito Capital – Centro de Gestión Industrial | Julio de 2022 |
| Andrés Felipe Velandia Espitia | Asesor Metodológico | Regional Distrito Capital - Centro de Diseño y Metrología | Agosto de 2022 |
| Rafael Neftalí Lizcano Reyes | Responsable Equipo desarrollo curricular | Regional Santander - Centro Industrial del Diseño y la Manufactura | Agosto de 2022 |
| Jhon Jairo Rodríguez Pérez | Corrector de estilo | Regional Distrito Capital - Centro de Diseño y Metrología | Septiembre de 2022 |
|  |  |  |  |

1. **CONTROL DE CAMBIOS**

**(Diligenciar únicamente si realiza ajustes a la Unidad Temática)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Nombre | Cargo | Dependencia | Fecha | Razón del Cambio |
| Autor (es) |  |  |  |  |  |