Universidad Continental

Práctica de construcción de software

TEMA: Programación por pares

III. Actividades

Usted y su par implementaran un proyecto en Python para cambiar un numero natural (z >= 0) a su equivalente en base b (2 <= b <= 16) utilizando la programación por pares, el flujo de trabajo Gitflow, el desarrollo guiado por pruebas y el lenguaje de programación Python.

Ejemplo, si el usuario introduce z=984 y b=16, entonces el programa mostrará la leyenda "3D8 representa al entero 984 en base 16".

Las tareas siguientes son para que pueda trabajar junto con su socio. Durante el ejercicio, asegúrese que usted cambie los roles a menudo; se sugiere cambiar cada diez minutos.

3.1. Diseñe los casos de prueba para el caso presentado.

1. Caso básico con valores pequeños:

z = 10, b = 2

z = 15, b = 8

z = 16, b = 16

2. Caso de número natural grande:

z = 984, b = 16

3. Caso con el límite inferior de la base:

z = 5, b = 2

4. Caso con el límite superior de la base:

z = 1000, b = 16

5. Caso con un número natural igual a cero:

z = 0, b = 8

6. Caso con el límite superior de z:

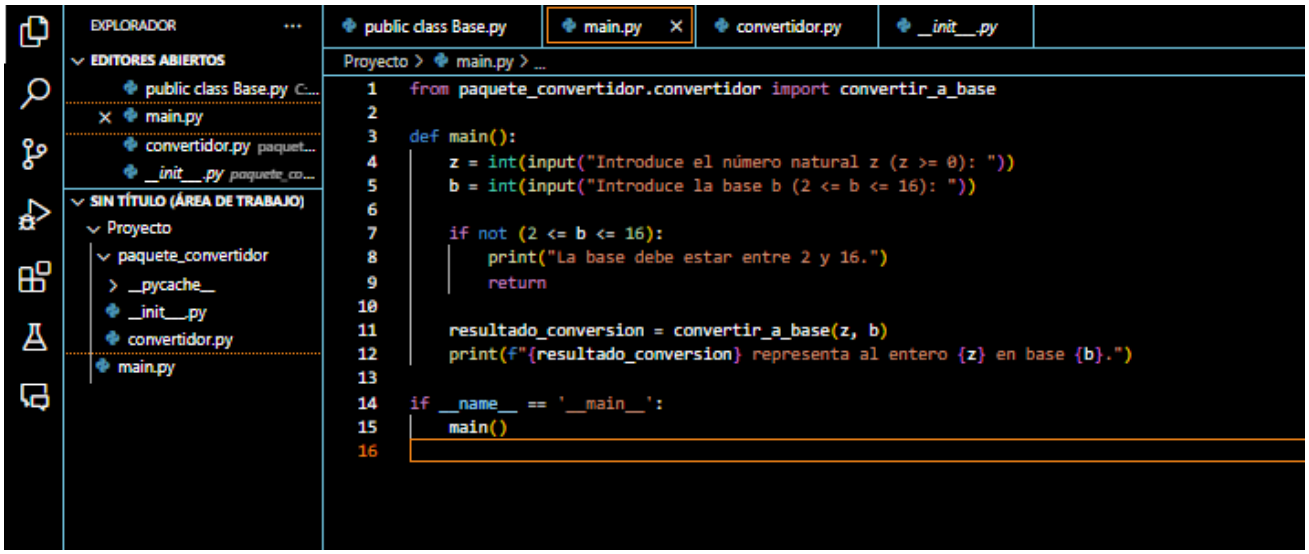
z = 9999, b = 2

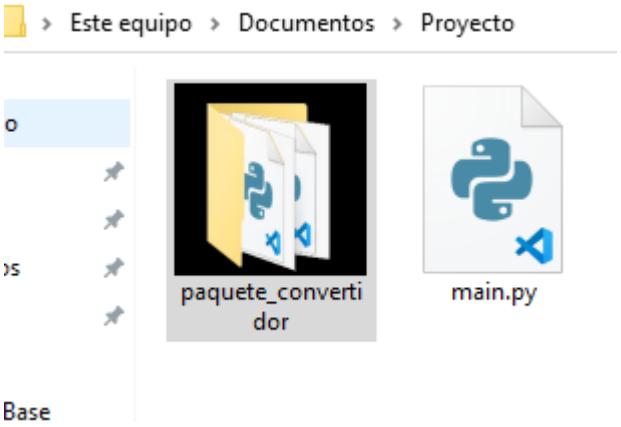
3.2. Si es necesario revise el marco teórico sobre programación por pares.

```
def convertir_a_base(z, b):
    if z == 0:
        return '0'
```

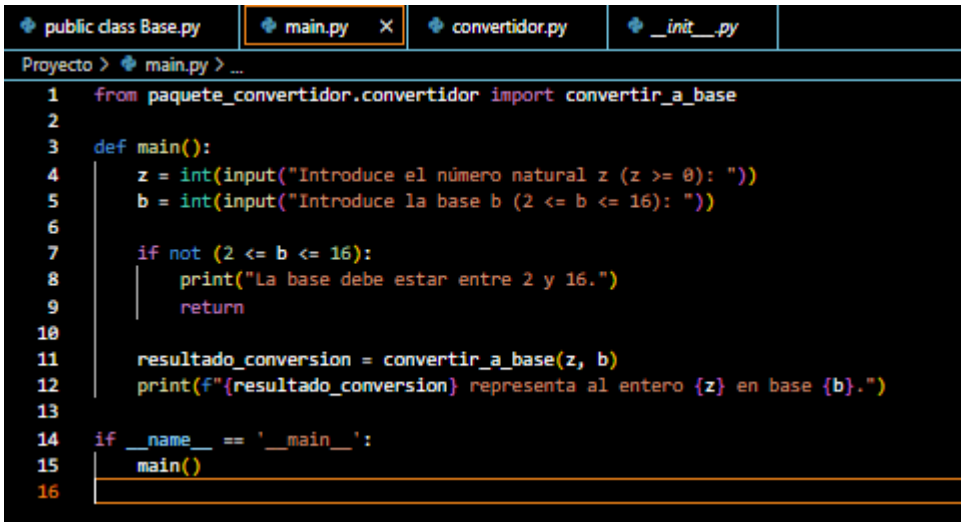
```
    digitos = "0123456789ABCDEF"

    resultado = ""
    while z > 0:
        resultado = digitos[z % b] + resultado
        z //= b

    return resultado

def main():
    z = int(input("Introduce el número natural z (z >= 0): "))
    b = int(input("Introduce la base b (2 <= b <= 16): "))

    if not (2 <= b <= 16):
        print("La base debe estar entre 2 y 16.")
        return

    resultado_conversion = convertir_a_base(z, b)
    print(f"{resultado_conversion} representa al entero {z} en base {b}.")

if __name__ == '__main__':
    main()
```

3.3. Organice el proyecto usando paquetes.





3.4. Divida la implementación en partes que sean fáciles de implementar, use los conceptos de modularidad y reutilización.



```python
def convertir_a_base(z, b):
    if z == 0:
        return '0'

    digitos = "0123456789ABCDEF"

    resultado = ""
    while z > 0:
        resultado = digitos[z % b] + resultado
        z //= b

    return resultado
```

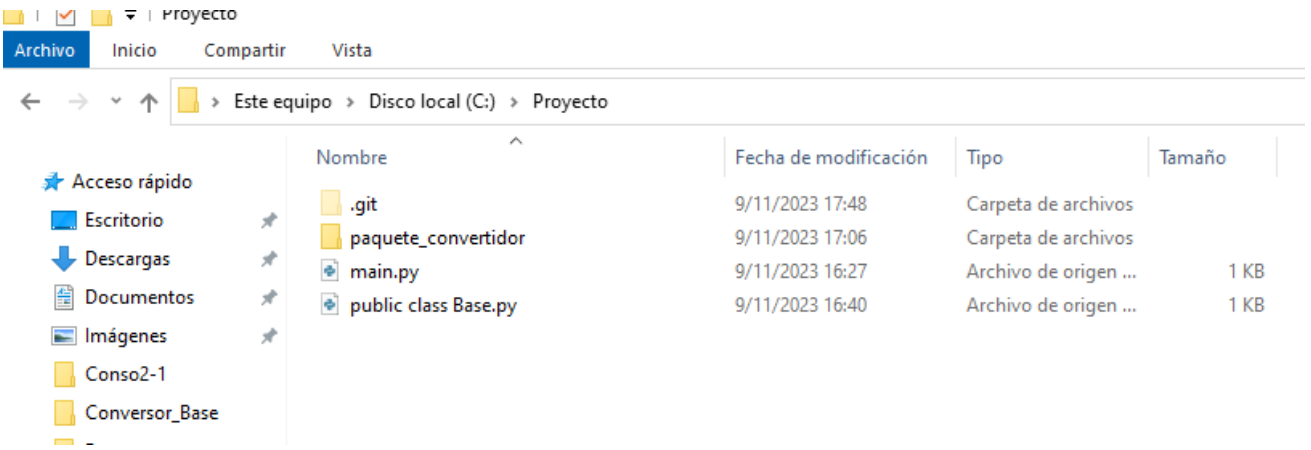
3.5. Cree el repositorio local (Git) y el remoto (GitHub).

En el github

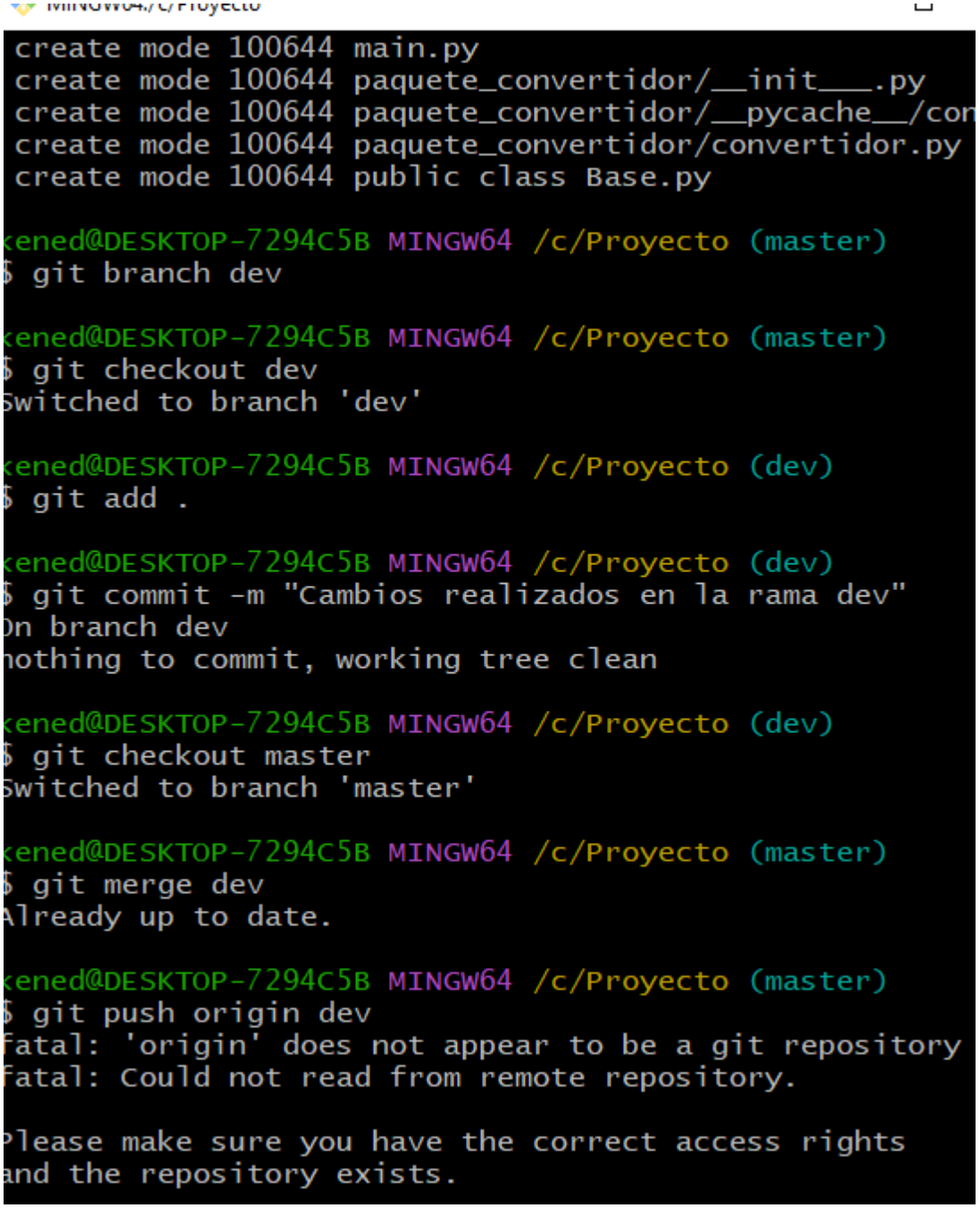
3.6. Realice la presentación del proyecto usando el archivo Readme.MD. La presentación debe incluir: el titulo del proyecto, los objetivos del proyecto, las lista de integrantes del equipo de desarrollo y un diagrama de HIPO (Hierarchical Input Process Output, enlace) de la implementación.

En el github

3.7. En el repositorio local, crea las ramas dev y master, y sincronízalo con el repositorio remoto

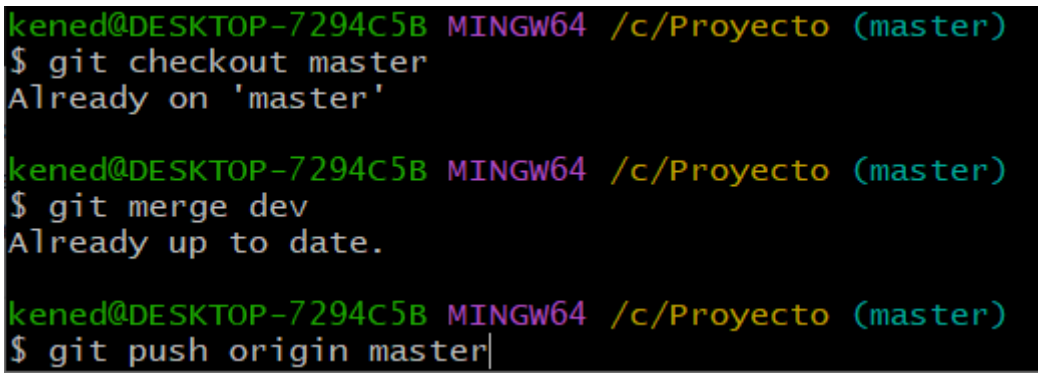


3.8. En el repositorio local, crea una rama para cada funcionalidad o feature e implemente la funcionalidad aplicando TDD.



3.8.1. Una vez probada la funcionalidad (para nuestro caso es la definición de terminado), sincronice la rama feature con el repositorio remoto. No elimine la rama feature.

3.8.2. Realice un merge de la rama implementada con las ramas dev y master. 3.9.



Comparta la ruta URL del proyecto en GitHub: https://forms.gle/KFBjUQ5euLaPPEkDA.