Week6:

```
+ __pycache__/
+ *.py[cod]
+ *$py.class
+
+ # C extensions
+ *.so
+
+ # Distribution / packaging
+ .Python
+ build/
+ develop-eggs/
+ dist/
+ downloads/
+ eggs/
+ .eggs/
+ lib/
+ lib64/
+ parts/
+ sdist/
+ var/
+ wheels/
+ *.egg-info/
+ .installed.cfg
+ *.egg
+ MANIFEST
+
+ # PyInstaller
+ # Usually these files are written by a python script from a template
+ # before PyInstaller builds the exe, so as to inject date/other infos into it.
+ *.manifest
+ *.spec
+
+ # Installer logs
+ pip-log.txt
+ pip-delete-this-directory.txt
+
+ # Unit test / coverage reports
+ htmlcov/
+ .tox/
+ .coverage
+ .coverage.*
+ .cache
+ nosetests.xml
+ coverage.xml
+ *.cover
+ .hypothesis/
+ .pytest_cache/
+
+ # Translations
+ *.mo
+ *.pot
+
+ # Django stuff:
+ *.log
+ local_settings.py
+ db.sqlite3
+
+ # Flask stuff:
+ instance/
+ .webassets-cache
+
+ # Scrapy stuff:
+ .scrapy
+
+ # Sphinx documentation
+ docs/_build/
+
+ # PyBuilder
+ target/
+
+ # Jupyter Notebook
+ .ipynb_checkpoints
+
+ # pyenv
+ .python-version
+
+ # celery beat schedule file
+ celerybeat-schedule
+
+ # SageMath parsed files
+ *.sage.py
```

```
+
+ # Environments
+ .env
+ .venv
+ env/
+ venv/
+ ENV/
+ env.bak/
+ venv.bak/
+
+ # Spyder project settings
+ .spyderproject
+ .spyproject
+
+ # Rope project settings
+ .ropeproject
+
+ # mkdocs documentation
+ /site
+
+ # mypy
+ .mypy_cache/
```

循环结构：

for 循环

语法：for 变量 in 可迭代对象:

特点：用于已知迭代次数的循环

可迭代对象包括：字符串、列表、元组、字典、集合、range 对象等

示例：

```
# 打印 1-5 的平方 for i in range(1, 6):
    print(f"{i}的平方是{i**2}")
```

range()函数：

range(stop)：0 到 stop-1

range(start, stop)：start 到 stop-1

range(start, stop, step)：start 到 stop-1，步长为 step

---

while 循环

语法：

while 条件表达式:

特点：用于条件满足时的循环

必须确保循环条件最终能变为 False，否则会无限循环

示例：

```
# 倒计时程序
count = 5while count > 0:
    print(count)
    count -= 1print("发射!")
```

循环控制语句

语句	作用	示例
break	立即退出当前循环	while True: if condition: break
continue	跳过当前迭代，进入下一次循环	for i in range(5): if i==2: continue

语句	作用	示例
else	循环正常结束后执行（非 break 退出）	for i in range(3): ... else: print("完成")

流程图：

复制

```
开始循环 → 检查条件 → True → 执行循环体 → 遇到 continue? → 是 → 下一轮循环
                                          ↓否
                                        遇到 break? → 是 → 退出循环
                                          ↓否
                                      循环结束 → 执行 else 块
```

条件分支

if语句

语法：

```
if 条件表达式:
```

示例：

```
age = 20if age >= 18:
    print("您已成年")
```

---

if-else 语句

语法：

```
if 条件表达式:
    # 条件为真时执行 else:
    # 条件为假时执行
```

示例：

```
num = 7if num % 2 == 0:
    print("偶数")else:
    print("奇数")
```

---

if-elif-else 语句

语法：

```
if 条件 1:
    # 代码块 1elif 条件 2:
```

    # 代码块 2...else:

    # 默认代码块

特点：

从上到下依次检查条件

第一个满足的条件对应的代码块被执行

else 是可选的

示例：

```
score = 85if score >= 90:
    grade = 'A'elif score >= 80:
    grade = 'B'elif score >= 70:
    grade = 'C'else:
    grade = 'D'print(f"成绩等级: {grade}")
```

异常处理

try-except

语法：try:

    # 可能出错的代码 except 异常类型:
    # 异常处理代码

示例：

```
try:
    result = 10 / 0except ZeroDivisionError:
    print("不能除以零!")
```

---

try-except-else

语法：

```
try:
    # 可能出错的代码 except 异常类型:
    # 异常处理 else:
    # 无异常时执行
```

示例：

```
try:
    num = int(input("请输入数字: "))except ValueError:
    print("输入的不是数字!")else:
    print(f"你输入的是: {num}")
```

try-finally

语法：

```
try:
    # 可能出错的代码 finally:
    # 无论是否异常都会执行
```

特点：

常用于资源清理（如关闭文件）

即使有 return 语句也会执行

示例：try:

```
    file = open("test.txt", "r")
    content = file.read()finally:
    file.close()   # 确保文件被关闭
```

综合练习

猜数字游戏：import random

```
target = random.randint(1, 100)
attempts = 0
print("猜数字游戏(1-100)，你有 5 次机会")
while attempts < 5:
    try:
        guess = int(input("请输入你的猜测: "))
        if guess < 1 or guess > 100:
            print("请输入 1-100 之间的数字!")
            continue

        attempts += 1


        if guess < target:
            print("猜小了!")
        elif guess > target:
            print("猜大了!")
        else:
            print(f"恭喜！你在{attempts}次内猜对了!")
```

```
                break
        except  ValueError:
            print("请输入有效的数字!")else:
        print(f"游戏结束!  正确答案是{target}")
```

代码解析：

使用while 循环控制游戏次数

try-except 处理非数字输入

if-elif-else 判断猜测结果

break 在猜对时提前退出循环

while-else 处理猜错所有机会的情况

总结表格

语句类型	关键字	使用场景
循环	for	已知迭代次数
循环	while	条件满足时循环
循环控制	break	立即退出循环
循环控制	continue	跳过当前迭代
条件分支	if	单条件判断
条件分支	if-else	二选一分支
条件分支	if-elif-else	多条件分支
异常处理	try-except	捕获并处理异常
异常处理	try-finally	确保清理代码执行
异常处理	raise	主动抛出异常