1、顺序结构：购买一支笔，价格是 2.5，数量两支笔，总额是 20 元，根据所给题目输出顺序结构代码

价格=price

数量=quantity

金额=paymet

```
# 输入数据
price = float(input("请输入商品单价："))  # 用户输入商品的单价，并转换为浮点数
quantity = int(input("请输入购买数量："))  # 用户输入购买的数量，并转换为整数
payment = float(input("请输入实际支付金额："))  # 用户输入实际支付的金额，并转换为浮点数

# 计算总价和找零
total = price * quantity  # 计算商品的总价，即单价乘以数量
change = payment - total  # 计算应找零的金额，即实际支付金额减去商品总价

# 输出结果（保留两位小数）
print(f"商品总价：{total:.2f}元")  # 输出商品总价，格式化为两位小数
print(f"应找零钱：{change:.2f}元")  # 输出应找零的金额，格式化为两位小数
```

输入数据：

price = float(input("请输入商品单价："))：提示用户输入商品的单价，使用 float()函数将输入的字符串转换为浮点数，以便进行小数计算。

quantity = int(input("请输入购买数量："))：提示用户输入购买的数量，使用 int()函数将输入的字符串转换为整数。

payment = float(input("请输入实际支付金额："))：提示用户输入实际支付的金额，使用 float()函数将输入的字符串转换为浮点数。

计算总价和找零：

total = price * quantity：计算商品的总价，即单价乘以数量。

change = payment - total：计算应找零的金额，即实际支付金额减去商品总价。如果 change 为负数，表示支付金额不足。

输出结果（保留两位小数）：

print(f"商品总价：{total:.2f}元")：使用格式化字符串输出商品总价，:.2f 表示格式化为两位小数。

print(f"应找零钱：{change:.2f}元")：使用格式化字符串输出应找零的金额，:.2f 表示格式化为两位小数。

2、单分支：假如一个人去上网，需要什么条件才能去上网—（需成年->=18 岁）

语法格式：

if 条件表达式：

执行语句

```
#判断一个人是否成年
age=20
if age>=18:
  print("你已经成年了！")
#输出：你已经成年了！
```

3、双分支：给定一个-5，判断它是不是正数

语法格式：

if 条件表达式：

条件为真时执行语句

else：

条件为假时执行语句

```
#判断一个数是否为正数
num=-5
if num>0:
  print("这是一个正数。")
else:
  print("这不是一个正数。")
#输出：这不是一个正数。
```

4、多分支：假如一个学生的分数是 85 分，去判断这个学生的成绩等级（成绩-score 等级-grade）

```python
score = 85  # 学生成绩

# 判断成绩等级
if score >= 90:
    grade = "A"  # 优秀
elif score >= 80:
    grade = "B"  # 良好
elif score >= 70:
    grade = "C"  # 中等
elif score >= 60:
    grade = "D"  # 及格
else:
    grade = "F"  # 不及格

print(f"学生的成绩是 {score}，等级是 {grade}")
```

使用 if-elif-else 结构，从高到低依次判断成绩范围。
每个成绩范围对应一个等级：
90-100：A（优秀）
80-89：B（良好）
70-79：C（中等）
60-69：D（及格）
低于 60：F（不及格）
根据输入的成绩，输出对应的等级

5、For 循环：计算 1-10 的总和

```python
# 初始化总和为0
sum = 0

# 使用for循环遍历1到10的整数
for i in range(1, 11):
    sum += i  # 将当前整数加到总和上

# 输出总和
print(f"1到10的总和是 {sum}")
```

使用 range(1, 11)生成从 1 到 10 的整数序列。
在 for 循环中，将每个整数 i 加到变量 sum 上。
循环结束后，sum 变量中存储的就是 1 到 10 的总和。
最后，使用 print 函数输出总和。

6、While 循环：计算 1-10 的总和

```python
# 初始化变量
sum = 0  # 用于存储总和
i = 1    # 从1开始计数

# 使用while循环
while i <= 10:
    sum += i  # 将当前数字加到总和中
    i += 1    # 更新计数器，向下一个数字移动

# 输出结果
print(f"1到10的总和是 {sum}")
```

初始化变量：
sum 用于存储总和，初始值为 0。
i 用于计数，从 1 开始。
while 循环：
条件是 i <= 10，表示只要 i 小于等于 10，循环就继续执行。

在循环体中，将当前的 i 加到 sum 中。
然后通过 i += 1 将 i 的值增加 1，进入下一个循环。
循环结束：
当 i 大于 10 时，循环结束，此时 sum 中存储的就是 1 到 10 的总和。

7、循环嵌套：打印九九乘法表

```
# 外层循环控制行数
for i in range(1, 10):  # i从1到9
    # 内层循环控制每行的列数
    for j in range(1, i + 1):  # j从1到i
        # 打印乘法表的每一项，使用end=" "防止自动换行
        print(f"{j}x{i}={i * j}", end="  ")
    # 每完成一行后换行
    print()
```

外层循环：
for i in range(1, 10)：控制乘法表的行数，i 表示当前行的最大乘数。
内层循环：
for j in range(1, i + 1)：控制每行的列数，j 表示当前行的起始乘数。
每行的列数从 1 到 i，即第 i 行打印 i 个乘法表达式。
打印格式：
使用 print(f"{j}×{i}={i * j}", end=" ")，其中 f"{j}×{i}={i * j}"是格式化字符串，用于打印乘法表达式。
end=" "参数用于防止 print 函数自动换行，而是用两个空格分隔每个表达式。
换行：
每完成一行后，使用 print()单独调用一个空的 print 函数，实现换行。