Monday Group

The prices of the 5 products as well as the tax rate are constants. They can be defined in two possible ways: both are acceptable.

```
#define TAX_RATE 0.06
#define PRICE_PRODUCT_1 2.98
#define PRICE_PRODUCT_2 4.50
.....
.....

OR

int main(void)
{
    const double TAX_RATE = 0.06
    const double PRICE_PRODUCT_1 = 2.98
    const double PRICE_PRODUCT_2 = 4.50
    .....
    .....

}
```

Function 2, 3 and 4 may or may not use addresses / references. However, if the function is only returning a single value (as is the case here for function 2, 3 and 4) it is better to return the value directly rather than use an address / reference.

```
// Example of calling functions 2, 3 and 4 where the result is returned
// directly. This is the preferred approach

double calculateSubTotal(int quantity1, int quantity2, int quantity3,
     int quantity4, int quantity5);

double calculateTax(double subTotal, double taxRate);

double calculateTotal(double subTotal, double tax);
...
...

int main(void)
{
     double subTotal, tax, totalPrice;
    ....
    ....
     subTotal         =         calculateSubTotal(product1_quantity,
product2_quantity, product3_quantity,
          product4_quantity, product5_quantity);
```

```
      tax = calculateTax(subTotal, taxRate);

      totalPrice = calculateTotal(subTotal, tax);
     ....
     ....
}

// Example of calling functions 2, 3 and 4 where the result is returned
via a reference


void calculateSubTotal(int quantity1, int quantity2, int quantity3,
      int quantity4, int quantity5, double &subTotal);

void calculateTax(double subTotal, double taxRate, double &tax);

void calculateTotal(double subTotal, double tax, double &total);
...
...

int main(void)
{
    double subTotal = 0.0;
    double tax = 0.0;
    double totalPrice = 0.0;
    ....
    ....
      calculateSubTotal(product1_quantity,          product2_quantity,
product3_quantity,
           product4_quantity, product5_quantity, subTotal);

      calculateTax(subTotal, taxRate, tax);

      calculateTotal(subTotal, tax, totalPrice);
     ....
     ....
}
```

The `main()` function can call function 2, 3 and 4 directly (easiest). Alternatively, `main()` can call an intermediate function A, which in turn calls function 2, 3 and 4. In this case, function A must return the results (subtotal, tax, total price) to `main()`

Tuesday Group

Function 2 and 3 may or may not use addresses / references. However, if the function is only returning a single value (as is the case here for function 2 and 3) it is better to return the value directly rather than use an address / reference.

```
// Example of calling functions 2 and 3 where the result is returned
// directly. This is the preferred approach

double compute_total_interest(double loan, double annual_interest,
double num_month_payments);
double compute_monthly_payment(double loan, double total_interest, int
num_month_payments);
....

int main(void)
{
      double total_interest, monthly_payment;

      total_interest = compute_total_interest(loan, annual_interest,
num_month_payments);

      monthly_payment = compute_monthly_payment(loan, total_interest,
num_month_payments);
     ...........
}


// Example of calling functions 2 and 3 where the result is returned
via a reference

void compute_total_interest(double loan, double annual_interest,
double num_month_payments, double& total_interest);
void compute_monthly_payment(double loan, double total_interest, int
num_month_payments, double& monthly_payment);
....

int main(void)
{
      double total_interest = 0.0;
     double monthly_payment = 0.0;

      compute_total_interest(loan,                    annual_interest,
num_month_payments, total_interest);

      compute_monthly_payment(loan,                     total_interest,
num_month_payments, monthly_payment);
     ...........
}
```

Friday Group

The EPF and SOCSO rates for both employees and employers are constants. They can be defined in two possible ways: both are acceptable.

```
#define SOCSO_EMPLOYEE 14.75
#define SOCSO_EMPLOYER 51.56
#define EPF_EMPLOYEE 0.11
#define EPF_EMPLOYER 0.13

int main(void)
{
 .....
 .....

OR

int main(void)
{
    const double SOCSO_EMPLOYEE = 14.75
    const double SOCSO_EMPLOYER = 51.56
    const double EPF_EMPLOYEE = 0.11
    const double EPF_EMPLOYER = 0.13
    ....
    ....
}
```

Functions 1, 2 and 3 may or may not use addresses / references. However, if the function is only returning a single value (as is the case here for all 3 functions) it is better to return the value directly rather than use an address / reference.

```
// Example of calling functions 1 and 2 where the result is returned
// directly. This is the preferred approach

double read_salary();
double calculate_epf(double gross_pay, double rate);
....
....

int main(void)
{

      double gross_pay;
      double epf_employee_deduct, epf_employer_deduct;

      gross_pay = read_salary();
      epf_employee_deduct = calculate_epf(gross_pay, EPF_EMPLOYEE);
      epf_employer_deduct = calculate_epf(gross_pay, EPF_EMPLOYER);
```

```
    .....
    .....
}

// Example of calling functions 1 and 2 where the result is
// returned via a reference

void read_salary(double& gross_pay);
void calculate_epf(double gross_pay, double rate, double &deduct);
...
...

int main(void)
{

    double gross_pay = 0.0;
    double epf_employee_deduct = 0.0;
    double epf_employer_deduct = 0.0;

    read_salary(gross_pay);

     calculate_epf(gross_pay, EPF_EMPLOYEE, epf_employee_deduct);
     calculate_epf(gross_pay, EPF_EMPLOYER, epf_employer_deduct);

}
```