

Q1:

```
4 double calculateBondPrice(double FV, double c, int n, double ytm) {
5     double bondPrice = 0.0;
6     for (int i = 1; i <= n; ++i) {
7         bondPrice += c / std::pow(1 + ytm, i); // 將利息折現
8     }
9     bondPrice += FV / std::pow(1 + ytm, n); // 將面額折現
10    return bondPrice;
11 }
```

這個 function 主要是方便直接折現出債券的現值，在下面的函示中呼叫即可

```
13 double newYTM(double FV, double c, int n, double p) {
14     double epsilon = 0.0001;
15     double lower = 0.0;
16     double upper = 1.0;
17     double ytm = (lower + upper) / 2.0;
18     double diff = calculateBondPrice(FV, c, n, ytm) - p;
19
20     while (std::abs(diff) > epsilon) {
21         if (diff < 0) {
22             upper = ytm;
23         }
24         else {
25             lower = ytm;
26         }
27         ytm = (lower + upper) / 2.0;
28         diff = calculateBondPrice(FV, c, n, ytm) - p;
29     }
30     return ytm;
31 }
```

這個函示主要是要用 bisection 的方式求出他的殖利率，也就是用逼近法

```
33 int main() {
34     double FV; // Face value
35     double c; // 利息
36     int n; // 期數
37     double p; // 價格
38     double r; // Risk-free interest rate
39     scanf("%lf", FV);
40     scanf("%lf", c);
41     scanf("%d", n);
42     scanf("%lf", p);
43     scanf("%lf", r);
44
45     double ytm = newYTM(FV, c, n, p);
46     double yieldSpread = ytm - r;
47
48     std::cout << "Yield Spread (S): " << yieldSpread << std::endl;
49
50     return 0;
51 }
```

再拿算出來的東西間掉無風險利率則可以符合題目要求

Q2:

由於程式內容與範例大致相似所以我將提供 terminal 以供查證

```
利率的輸入及定存的FV的計算
```

```
IntFix[0] = 0.001229
```

```
IntFix[1] = 0.003688
```

```
IntFix[2] = 0.007375
```

```
IntFix[3] = 0.014750
```

```
IntFix[4] = 0.029500
```

```
一個月定存FV: 1.001229
```

```
三個月定存FV: 1.003688
```

```
六個月定存FV: 1.007375
```

```
一年定存FV: 1.014750
```

```
兩年定存FV: 1.029500
```