
Computer Programming: Skills & Concepts (CP1)

Case Study: Coin change

15th October 2009



Coin Change

We want to write a program that

- ask the user for an amount of money
- calculates the coins needed for this amount
- outputs the number of each coin

Type of Coins

Coins range from 1p to £2

```
const int
```

```
    C1 = 200, C2 = 100, C3 = 50, C4 = 20,  
    C5 = 10,  C6 = 5,  C7 = 2,  C8 = 1;
```

Three functions

```
if ( ReadInput(&amount) != EXIT_SUCCESS) {
    printf("Failure in ReadInput\n");
    return EXIT_FAILURE;
}
if ( CalculateCoins(amount) != EXIT_SUCCESS) {
    printf("Failure in CalculateCoins\n");
    return EXIT_FAILURE;
}
if ( PrintResult(amount) != EXIT_SUCCESS) {
    printf("Failure in PrintResult\n");
    return EXIT_FAILURE;
}
```

Take Input from User

```
int input = 0 ;  
printf("Enter the amount (in pence) to be returned  
to the user: ");  
scanf("%d", &input);
```

Take Input from User (Error Tolerant)

```
int input = 0 ;

do {
    printf("Enter the amount (in pence) to be returned
           to the user: ");
    while (scanf("%d", &input) !=1) {
        scanf("%*s") ;
        printf("That wasn't a number - please try again: ");
    }
} while (input < 0 ) ;
```

Take Input from User (Full Function)

```
int ReadInput(int *amount) {
    int input = 0 ;
    do {
        printf("Enter the amount (in pence) to be returned
                to the user: ");
        while (scanf("%d", &input) !=1) { scanf("%*s") ;
            printf("That wasn't a number - please try again: ");
        }
    } while (input < 0 ) ;
    *amount = input ;    // Set the value of amount to equal input
    return EXIT_SUCCESS ;
}
```

Calculate Coins

```
int pot = 0; // Total value of coins so far selected.
while (pot < amount) {
    if (pot + C1 <= amount) {
        pot += C1; ++n1;
    } else if (pot + C2 <= amount) {
        pot += C2; ++n2;
    } else if (pot + C3 <= amount) {
        pot += C3; ++n3;
    } else if (pot + C4 <= amount) {
        pot += C4; ++n4;
    } else if (pot + C5 <= amount) {
        pot += C5; ++n5;
    } else if (pot + C6 <= amount) {
```

```
    pot += C6;  ++n6;
} else if (pot + C7 <= amount) {
    pot += C7;  ++n7;
} else {
    /* pot + C8 <= amount.  (Why?) */
    pot += C8;  ++n8;
}
}
```

Catching Programming mistakes

```
assert(  
    n1*C1 + n2*C2 + n3*C3 + n4*C4 + n5*C5 + n6*C6 + n7*C7 + n8*C8 ==  
    && pot <= amount  
); // If the argument is true, call break
```

If `assert(expression)` is false, the program stops with an error message.

Output to User

```
int PrintResult(int amount) {
    printf("%dp may be returned using the following
           combination of coins:\n", amount);
    if (n1) printf("%4d %dp coins,\n", n1, C1);
    if (n2) printf("%4d %dp coins,\n", n2, C2);
    if (n3) printf("%4d %dp coins,\n", n3, C3);
    if (n4) printf("%4d %dp coins,\n", n4, C4);
    if (n5) printf("%4d %dp coins,\n", n5, C5);
    if (n6) printf("%4d %dp coins,\n", n6, C6);
    if (n7) printf("%4d %dp coins,\n", n7, C7);
    if (n8) printf("%4d %dp coins.\n", n8, C8);
    return EXIT_SUCCESS;
}
```