

Oct 17, 04 17:56

coins_funcs.c

Page 1/3

```

/*
 * A program to compute a small set of coins that sum to a given amount.
 * The program illustrates the 'greedy heuristic,' which happens,
 * for this choice of coins, to yield an optimal solution
 * (i.e., one using the smallest number of coins).
 */

#include <stdlib.h>
#include <stdio.h>
#include <assert.h>

/* Function prototypes */

int ReadInput(int *amount) ;
int CalculateCoins(int amount) ;
int PrintResult(int amount) ;

/* globals */

const int
/* Denominations of coins currently in circulation. */
C1 = 200, C2 = 100, C3 = 50, C4 = 20,
C5 = 10, C6 = 5, C7 = 2, C8 = 1;

int n1, n2, n3, n4, n5, n6, n7, n8;
/* Number of coins of each denomination */

int main(void)
{
    int amount = 0; /* Amount to be returned to the user. */

    n1 = n2 = n3 = n4 = n5 = n6 = n7 = n8 = 0;

    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;
    }

    return EXIT_SUCCESS;
}

/* Function to take input from the user */

int ReadInput(int *amount) {
    int rtn = 0; /* Important to initialise this.

```

Oct 17, 04 17:56

coins_funcs.c

Page 2/3

```

int input = 0 ;

do {

    printf("Enter the amount (in pence) to be returned to the user: ");

    while (scanf("%d", &input) !=1) {
        scanf("%s");
        printf("Enter the amount (in pence) to be returned to the user: ");
    }

} while (input < 0) ;

*amount = input ; /* Set the value of amount to equal input
return EXIT_SUCCESS ;
}

/* An algorithm to compute the smallest number of coins to return
to the user to equal the parameter amount. */

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

/* Function to print results to the user */

int PrintResult(int amount) {
    printf("%d 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);

```

Oct 17, 04 17:56

coins_funcs.c

Page 3/3

```
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;  
}
```