

Part C - Donna Wright

by Donna Wright

FILE	11968_DONNA_WRIGHT_PART_C_-_DONNA_WRIGHT_173926_563914631.DOCX (107.85K)		
TIME SUBMITTED	02-MAY-2019 02:10PM (UTC+0100)	WORD COUNT	875
SUBMISSION ID	106100629	CHARACTER COUNT	4357

Programs in C and documentation

Part C
By Donna Wright

Contents

Part C (Implementation)	3
Function based exercise: Pairs.....	3
1.1 Problem breakdown.....	3
1.1.1 Generate and mask random pairs	3
1.1.2 Allow user to select and compare cards . Error! Bookmark not defined.	
1.1.3 Hard mode.....	3
1.1.4 Record amount of moves taken..... Error! Bookmark not defined.	
1.2 Annotated code.....	4
1.3 Testing	6
1.3.1 Generate and shuffle list of words.....	6
References.....	10

Part C (Implementation)

Function based exercise: Pairs.

1.1 Problem breakdown

The purpose of this exercise is to build a card matching game, allowing a player to try and find matching pairs in as few moves as possible. The player will initially be presented with six pairs which will be shuffled. The player should be allowed to reveal the identity of two items at a time in order to locate all pairs.

1.1.1 Generate and mask random pairs

In this stage the pairs need to be declared and their placement randomised. The shuffled placement should not be visible to a player. In order to generate random numbers in C, two additional libraries are required. These are `<stdlib.h>` and `<time.h>`.

1. Add additional libraries to code
2. Create array of numbers
3. Use `()swap` and `()randomize` to shuffle numbers

1.1.2 Invite user to play

This stage requires input from the user

1. Print a 4x3 table of letters
2. Ask user to select the letter of the first card they wish to turn
3. Store players choice
4. Ask player to select the second letter they wish to turn
5. Store players choice
6. Display random numbers linked to the letters chosen and advise if a match

1.1.3 Continuous play

Player needs to be allowed to continue to select cards with an aim to find all pairs

1. Create while loop to allow player to keep playing for as long as they want
2. In each loop allow player should be able to select 2 cards to compare
3. In each loop print cards selected
4. Advise player whether their turn resulted in a match
5. Ask player if they wish to continue
6. Continue until all cards matched or player chooses to end the game

1.1.4 Record turns and advise of matches *//I was unable to code this stage*

This stage would look to 'turn the cards back', record the number of turns and advise of matches so far. It would also end the game when all matches have been found.

1. Clear screen if player chooses to continue
2. Record number of turns taken
3. Record matches made

1.2 Annotated code

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

void swap(int *a, int *b)
{
    int temp = *a;
    *a = *b;
    *b = temp;
}

void randomize(int pairs[], int n)
{
    srand(time(NULL));
    int i;
    for (i = n - 1; i > 0; i--)
    {
        int j = rand() % (i + 1);
        swap(&pairs[i], &pairs[j]);
    }
}

void main()
{
    int pairs[12] = {100,200,300,400,500,600,100,200,300,400,500,600};
    int card1, card2;
    int n = sizeof(pairs) / sizeof(pairs[0]);
    int i;
    char choice;

    randomize(pairs, n);

    printf("The aim of the game is to find the pairs\nThe following values are
hidden behind the cards:\n");
    printf("100\n200\n300\n400\n500\n600\n\nDo you want to play?\n");
    scanf(" %c", &choice);

    while (choice == 'y' || choice == 'Y') //both uppercase and lowercase covered
    {
        printf("  0  1  2  3\n");
        printf("  4  5  6  7\n");
        printf("  8  9 10 11\n\n");

        printf("Enter the number of the first card you would like to turn: ");
        scanf("%d", &card1);
        printf("Enter number of the second card you would like to turn: ");
        scanf("%d", &card2);

        if (pairs[card1] == pairs[card2])
        {
            printf("You found %d - %d\n", pairs[card1], pairs[card2]);
            printf("It's a match!\n");
            printf("Do you wish to continue?\n");
            scanf(" %c", &choice);
        }
        else
        {
            printf("You found %d - %d\n", pairs[card1], pairs[card2]);
```

```

        printf("Unlucky, that's not a match\n");
        printf("Do you wish to continue?\n");
        scanf(" %c", &choice);

    }
    if (choice == 'n' || choice == 'N') //both uppercase and lowercase covered
    {
        printf("\nYou're no fun!\n");
    }
}
//list all cards after sshuffle
printf("Thanks for playing, the cards are revealed below:\n");
printf("    %d %d %d %d\n", pairs[0],pairs[1],pairs[2],pairs[3]);
printf("    %d %d %d %d\n", pairs[4], pairs[5], pairs[6], pairs[7]);
printf("    %d %d %d %d\n", pairs[8], pairs[9], pairs[10], pairs[11]);
}

```

1.3 Testing

1.3.1 Generate and shuffle list of words

Program shuffles the declared array of words to random locations.

```
Thanks for playing, the cards are revealed below:  
100 500 100 400  
400 200 300 600  
200 500 600 300
```

```
Thanks for playing, the cards are revealed below:  
500 400 200 600  
400 100 300 200  
100 300 500 600
```

Result: program ran as expected.

Player given result of turn:

```
Enter number of the second card you want to turn:  
You found 400 - 600  
Unlucky, that's not a match
```

```
Enter number of the second card you want to turn:  
You found 400 - 400  
It's a match!
```

Result: Program ran as expected.

1.3.2 Invitation to play

Player is invited to play and prompted to enter card selections. Following this they are informed whether selection is a match and selected 'cards' revealed to user.

Invite to play (y entered by player):

```
The aim of the game is to find the pairs  
The following values are hidden behind the cards:  
100  
200  
300  
400  
500  
600  
  
Do you want to play?  
y  
0 1 2 3  
4 5 6 7  
8 9 10 11  
  
Enter the number of the first card you would like to turn:
```

Result: program ran as expected.

Invite to play (n entered by player):

```
The aim of the game is to find the pairs
The following values are hidden behind the cards:
100
200
300
400
500
600

Do you want to play?
n
Thanks for playing, the cards are revealed below:
100  400  600  300
600  500  500  200
300  100  400  200
```

Result: program ran as expected.

Invite to play (invalid entry by player)

```
The aim of the game is to find the pairs
The following values are hidden behind the cards:
100
200
300
400
500
600

Do you want to play?
4
Thanks for playing, the cards are revealed below:
300  500  400  200
100  500  400  300
600  600  200  100
```

Result: program did not run as expected. Whilst game ended, invalid entry did not come up

User enters valid 'choices':

```
Enter the number of the first card you would like to turn: 1
Enter number of the second card you would like to turn: 2
You found 400 - 100
Unlucky, that's not a match
Do you wish to continue?
```

Result: program ran as expected.

Player enters invalid 'choices':

```
Enter the number of the first card you would like to turn: e
Enter number of the second card you would like to turn:
C:\Users\user-admin\Desktop\Uni work\ACCA4017 - Software Engineering\
ss 1444) exited with code -1073741819.
```

Result: program failed.

1.3.3 Continuous play

Player advised whether they found a match and asked if they wish to continue:

```
Enter the number of the first card you would like to turn: 0
Enter number of the second card you would like to turn: 1
You found 300 - 600
Unlucky, that's not a match
Do you wish to continue?
y
  0  1  2  3
  4  5  6  7
  8  9 10 11

Enter the number of the first card you would like to turn: 0
Enter number of the second card you would like to turn: 2
You found 300 - 200
Unlucky, that's not a match
Do you wish to continue?
y
  0  1  2  3
  4  5  6  7
  8  9 10 11

Enter the number of the first card you would like to turn: 0
Enter number of the second card you would like to turn: 3
You found 300 - 300
It's a match!
Do you wish to continue?
```

Result: Program ran as expected.

User allowed to exit game:

```
Do you wish to continue?
n

You're no fun!
Thanks for playing, the cards are revealed below:
 300  600  200  300
 500  500  400  100
 200  100  600  400

C:\Users\user-admin\Desktop\Uni work\ACCA4017 - Soft
```

Result: Program ran as expected.

Player entered invalid entry:

```
Do you wish to continue?  
e  
Thanks for playing, the cards are revealed below:  
400 200 300 500  
600 100 400 300  
200 600 100 500  
C:\Users\user-admin\Desktop\Uni work\ACCA4017 - Software
```

Result: Program ended.

References

- [1] M. McGrath, C Programming, vol. 4th edition, Warwickshire: In easy steps, 2014.

Part C - Donna Wright

GRADEMARK REPORT

FINAL GRADE

/100

GENERAL COMMENTS

Instructor

PART C

Coding/Annotation- Be careful when looking things up - swap() is technically a C++ library function, not a C one. The swap function seems a little strange. I'm not entirely sure why n is needed as it's the same value every time anyway? You need some validation on the inputs for the cars - as it is the user could enter a value outside 0-11 (eg negative numbers) You also are not marking matches as matched to prevent them being matched in future turns (ie you could match the same positions over and over again). You also need to end the game when all pairs have been matched, not just when the user chooses to. 11.5/20

Documentation - It would be better to have all the stages in one set of numbering, then you could show which stage you go back to with a loop. Testing is OK, but make sure you are testing the full range of values (eg negative numbers as suggested above. 9/15

Total 20.5/35

Overall total for assignment: 45.5+20.5

66%

Test mark 77%

Overall mark for module 69%

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10
