```
#include "Dancer.h"
#include <iostream>
using namespace std;
Dancer::Dancer()
{
    id = 0;
    hours = 0;
    rate = 0;
}

void Dancer::read_val(ifstream &inputfile)
{
    inputfile >> id >> hours >> rate;
}

void Dancer::print_val(ofstream& outputfile)
{
        outputfile << id << " " << hours << " " << rate << endl;
}

void Dancer::sort_by_id(Dancer d[], int k)
{
    int temp;
    float tempr;
    for (int i = 0; i < 12; i++)
    {
        for (int j = i + 1; j < 12; j++)
        {
            if (d[i].id > d[j].id)
            {

                temp = d[i].id;
                d[i].id = d[j].id;
                d[j].id = temp;
                temp = d[i].hours;
                d[i].hours = d[j].hours;
                d[j].hours = temp;
                tempr = d[i].rate;
                d[i].rate = d[j].rate;
                d[j].rate = tempr;
            }
        }
    }
    for (int i = 0; i < 12; i++)
    {
        for (int j = i + 1; j < 12; j++)
        {
            if (d[i].id == d[j].id)
            {
```

```cpp
                if (d[i].hours < d[j].hours)
                {
                    temp = d[j].hours;
                    d[j].hours = d[i].hours;
                    d[i].hours = temp;
                }
            }
        }
    }
}


void Dancer::sort_by_rate(Dancer d[], int k)
{
    float tempr;
    for (int i = 0; i < 12; i++)
    {
        for (int j = i + 1; j < 12; j++)
        {
            if (d[i].id == d[j].id)
            {
                if (d[i].rate > d[j].rate)
                {
                    tempr = d[i].rate;
                    d[i].rate = d[j].rate;
                    d[j].rate = tempr;
                }
            }
        }
    }
}

int Dancer::couple_counter(Dancer d[], int k)
{
    int couples = 0;
    for (int i = 0; i < 12; i++)
    {
        couples++;
        if (d[i].id == d[i + 1].id)
            couples--;
    }
    return couples;
}

void Dancer::u_id(Dancer d[], ofstream& outputfile)
{
    int sponsers = 1;
    float total = 0;
    for (int i = 0; i < 12; i++)
    {
            if (d[i].id == d[i+1].id)
            {
                sponsers += 1;
                total += d[i].get_earnings(d, k:i);
            }
            else if (d[i].id != d[i + 1].id)
            {
                total += d[i].get_earnings(d, k:i);
                outputfile << d[i].id << ": " << sponsers << " Sponsers $ 
                  " << total << endl;
                sponsers = 1;
                total = 0;
            }
    }
}

float Dancer::get_earnings(Dancer d[], int k)
{
    return d[k].hours * d[k].rate;
}

int Dancer::get_id()
{
    return id;
}

int Dancer::get_hrs()
{
    return hours;
}

float Dancer::get_rate()
{
    return rate;
}

float Dancer::total_raised(Dancer d[])
{
    float total = 0;
    for (int i = 0; i < 12; i++)
    {
        total += d[i].hours * d[i].rate;
    }
     return total;
}

void Dancer::swap(int k, int j, Dancer d[])
{
```

```cpp

}

void Dancer::best_couple(Dancer d[], ofstream& outputfile)
{
    float best = 0;
    int bestsponsers;
    int sponsers = 1;
    int coupleid;
    float total = 0;

    for (int i = 0; i < 12; i++)
    {
        if (d[i].id == d[i + 1].id)
        {
            sponsers += 1;
            total += d[i].get_earnings(d, k:i);
        }
        else if (d[i].id != d[i + 1].id)
        {
            total += d[i].get_earnings(d, k:i);
            if (total > best)
            {
                best = total;
                coupleid = d[i].id;
                bestsponsers = sponsers;
                sponsers = 1;
                total = 0;
            }
            if (!(total > best))
            {
                sponsers = 1;
                total = 0;
            }
        }
    }

    outputfile << "Couple with the most money is : " << coupleid << " With " << bestsponsers << " sponsers, earning $" << best << endl;
}

```