Overview

We where asked to create a program which calculated the total cost of a wedding for 64 people. This meant the program had to calculate many individual costs for a variety of items and then sum the individual costs together at the end.

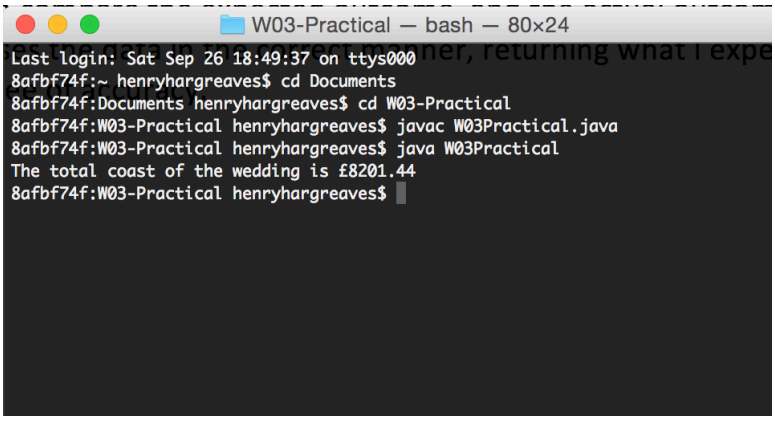
Design

The design of my program is simple and uses multiple different variables to store the different values, for each item. Each variable is stored as a float; this is because the float type allows decimal numbers to be stored in 32 bits, thus saving space as the double variable type required 64 bits. A variable named 'total' was used to store the result of the calculation which summed up all the individual costs. This value, to two decimal places, was then displayed at the end.
The extensions I added meant that the original structure of my program was largely altered. Rather than using numerous different variables for each item I decided to use 3 arrays. The first two arrays are arrays of float values and where called 'Prices' and 'Prices1'. These arrays store the cost of each item and the cost of each item from a rival wedding firm, respectively. The third array is an array of strings and was called 'Item'. This array is used to store the name of each item. The program could then print the contents of the arrays when ever a sub method was called. Users are able to view the contents of the arrays and edit the values stored in the 'Prices' array too. Finally, the user is able to enter the different quantity of each item from the array they require. The total of the wedding is then calculated based on the information input by the user.

Testing

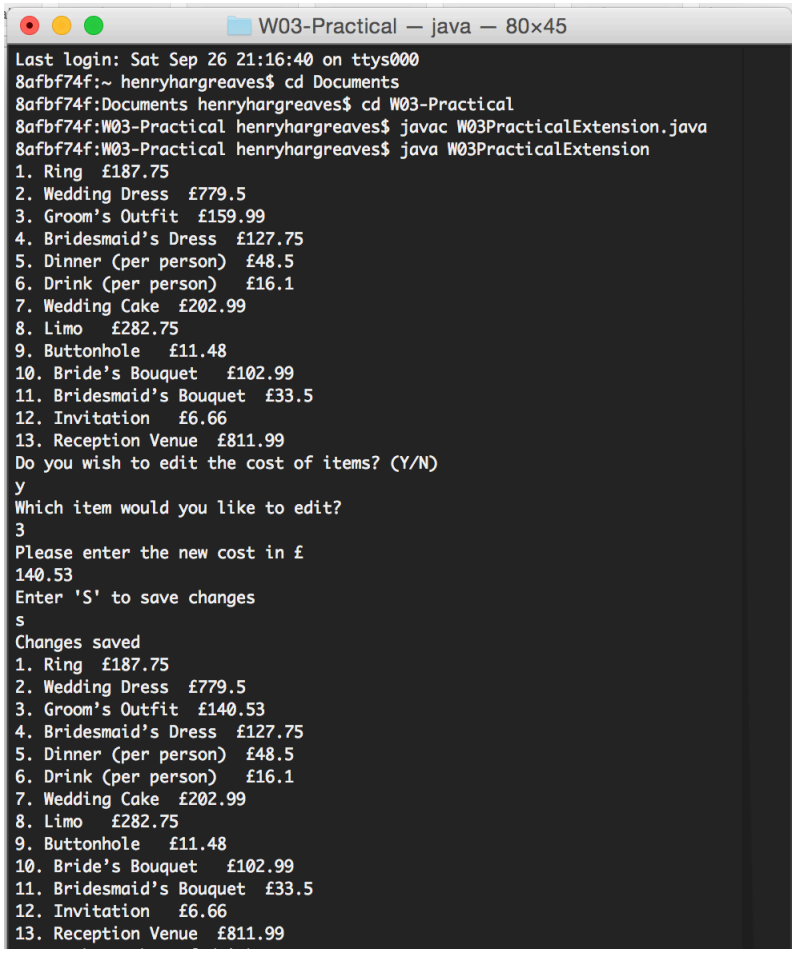
To test my program, I will calculate the values the program should output I will then compare these value to the actual values the program output. I will also make sure I have suitable test data when I am testing my extension; this will range from data that would expected by the program and may be commonly entered, as well a data that the program would not expect for example entering a letter, where an integer is expected. Entering invalid data to try and produce errors. This will help find bugs and glitches in my program. Again I can then compare the expected outcome, and the actual outcome. If the program executes and uses the data in the correct manner, returning what I expected, I will know it works to a degree of accuracy.

Examples

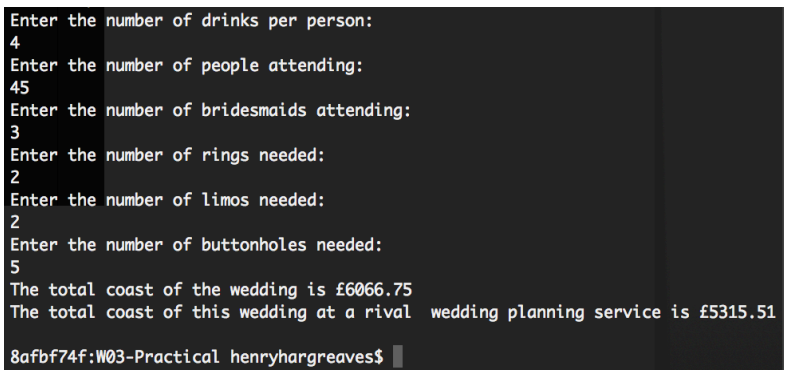


The total of the wedding was then correctly calculated.

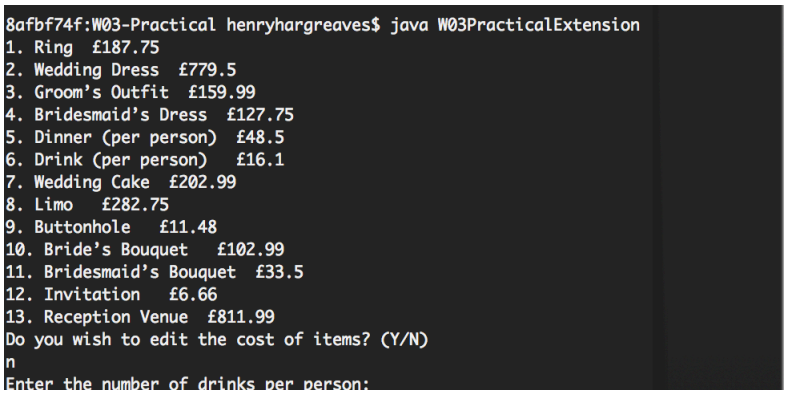
Extension



Here I ran the program through and edited the value of item number 3 in the array. As you can see the cost f this item was correctly updated.



The program continued and I was then able to able to enter the different quantity of each item required. The total of the wedding was then correctly calculated.



Here I ran the program through and edited the value of item number 3 in the array. As you can see the cost f this item was correctly updated.

```
Do you wish to edit the cost of items? (Y/N)
y
Which item would you like to edit?
2
Please enter the new cost in £
d
Make sure you enter a decimal number
753.65
Enter 'S' to save changes
f
Changes not saved
1. Ring  £187.75
2. Wedding Dress  £779.5
3. Groom's Outfit  £159.99
4. Bridesmaid's Dress  £127.75
5. Dinner (per person)  £48.5
6. Drink (per person)  £16.1
7. Wedding Cake  £202.99
8. Limo  £282.75
9. Buttonhole  £11.48
10. Bride's Bouquet  £102.99
11. Bridesmaid's Bouquet  £33.5
12. Invitation  £6.66
13. Reception Venue  £811.99
Enter the number of drinks per person:
```

Here I ran the program and purposefully entered invalid data.

- Entering 'd' for new cost
- Entering f when asked to enter 'S'

The program detected this and continued as it should.

Evaluation

Both my initial code and my extension worked as expected. The initial code was very limited in its functionality and was only able to complete one basic task. The user was unable to alter of the variables and thus not able to change any of the values used in computation. However the program did, as expected, calculate the cost of a wedding for 64 people.
Due to the limitations of my initial code I decided to make the extension far more flexible and interactive, offering users a greater range of functionality and generally extending the practicality of the program. The program was successfully able to calculate and offers a quote for rival wedding planners. The quotes where calculated using inputs from the user. The user was able to enter the costs of items and/or the number of each item required.

Conclusion

One of the first problems I had was: how to store the values of each item. This was quite a fundamental problem as there are multiple ways of storing the data, moreover another issue was whether or not to even store the data or just the raw values in the correct calculation. I decided the best solution was to declare multiple float type variables and then call the variables in the calculations, this made the program easier to follow and understand. However in my extension using numerous variables would have been unpractical and I had to look for an alternative method for storing data. The answer was to use an array which at first was hard to correctly implement and manipulate.
If I had more time I would have liked to create a GUI (graphical user interface) which would have made the program more user friendly and may have also resulted in, making the process of inputting and outputting data a quicker and more intuitive process.