

# Pick Me! The Application Selection Process Assistant

---

Jessica Irwin  
MBA 614

## Executive Summary

### Motivation

For the past two years I have had the opportunity to work as an accountant and program manager in the Mathematics Department. I work closely with a yearlong program called the IMPACT Program, which trains students in high-level research in mathematics and statistics ([www.impact.byu.edu](http://www.impact.byu.edu)). As the National Science Foundation provides funding, each student receives a \$10,000 scholarship to be a part of the program. The program is highly competitive and we receive over 30 well-qualified applications from students per year. However, we can accept only 10 students.

Over the past two years, I have spent many hours collecting and preparing the data received from applicants. Currently, I use a spreadsheet in GoogleDocs to store most of the data (some information is not stored there for security reasons), but I would like to present the data to the program directors in the most simplest and user-friendly way possible. I would also like to keep all application information in one secure place.

### My Solution

In order to assist in the selection process of our future year's cohort, I was able to use the tools within Excel and Visual Basic to streamline the collection of large amounts of information from the applicants. Through Visual Basic, I was able to extract data from the application (a Word document) and write that data into a worksheet. This will allow the user to save a lot of time and energy by not manually inputting each applicant's information. The data extracted from Word includes the applicant's name, contact information, GPA, gender, and other relevant information we use in making our decisions. User forms were created to search, view, and edit the applicant information, allowing for easy changes to the data if needed. I was also able to do some analysis on the group of applicants—such as rank the students in order of highest to lowest overall and major GPA. In addition, I automated the creation of an interview schedule, as well as creating a user form for the email of acceptance and rejection letters. I feel that I was able to create a program that could be adapted to other application and interview processes that occur in the Mathematics Department, and not only for the IMPACT Program.

## Implementation

Upon opening the Excel file, a user is greeted with the following view:



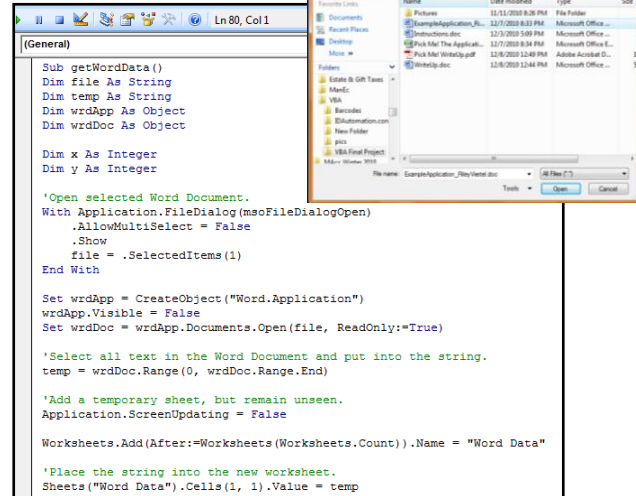
The following user form is displayed, allowing the user to select one of five options:

1. Import Data
2. Search
3. Applicant List
4. Email
5. Interviewing

In addition, a tab called “Applicant Statistics” has been added to the file. Each of these options and the functions they perform will be described in detail.

## Import Data

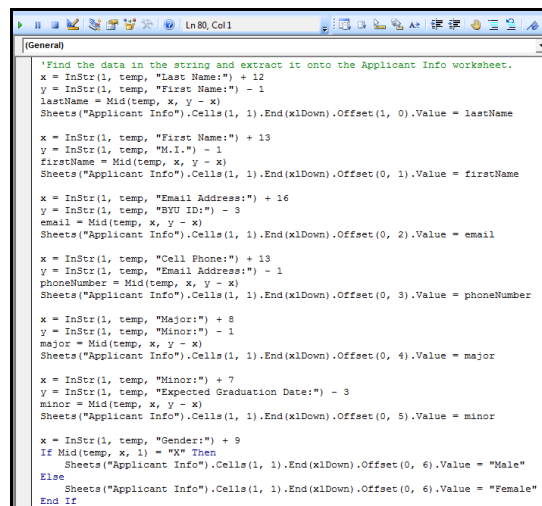
The import data function allows a user to extract data from a Word document and place it onto the “Applicant Info” sheet in the workbook. This eliminates the need for data entry of the applicant’s name, contact information, major/minor, gender, citizenship, and GPA. As the subprocedure runs, it opens up the Word document selected by the user. Next, it selects and copies all the data from the Word document, and pastes that information into a temporary worksheet in the Excel workbook.



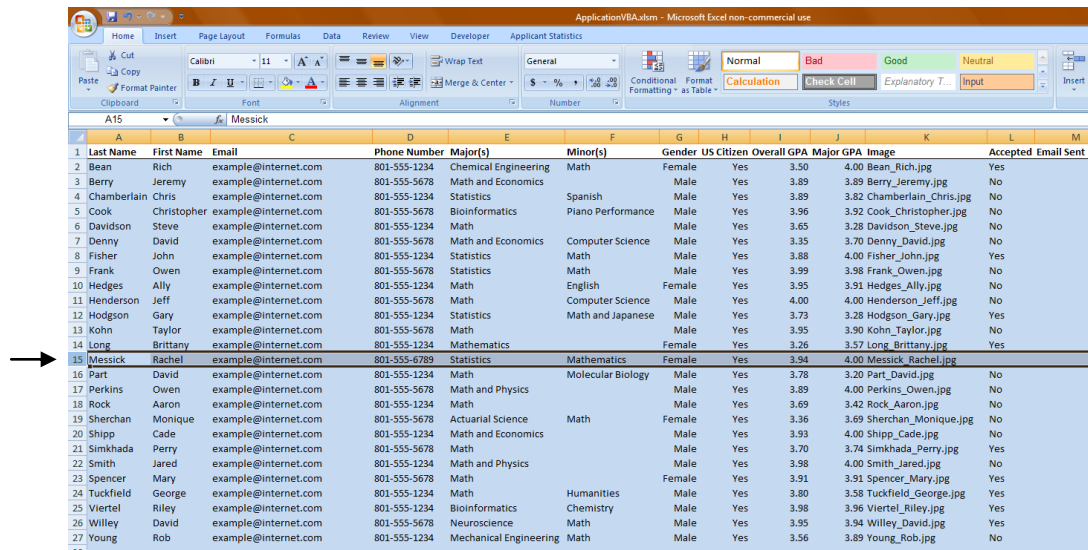
Then, the subprocedure extracts the data from the imported string using the “instring” function. The data extracted is placed onto first blank row in the “Applicant Info” sheet in its correct column. In this example, Rachel Messick’s application data was imported into the worksheet from her Word document application:

The image shows an Excel spreadsheet titled "ApplicationVBA.xlsx - Microsoft Excel - non-commercial use". The spreadsheet has columns A through M. The data is as follows:

1	Last Name	First Name	Email	Phone Number	Major(s)	Minor(s)	Gender	US Citizen	Overall GPA	Major GPA	Image	Accepted	Email Sent
2	Bean	Rich	example@internet.com	801-555-1234	Chemical Engineering	Math	Female	Yes	3.50	4.00	Bean_Rich.jpg	Yes	
3	Berry	Jeremy	example@internet.com	801-555-5678	Math and Economics		Male	Yes	3.89	3.89	Berry_Jeremy.jpg	No	
4	Chamberlain	Chris	example@internet.com	801-555-1234	Statistics	Spanish	Male	Yes	3.89	3.82	Chamberlain_Chris.jpg	No	
5	Cook	Christopher	example@internet.com	801-555-5678	Bioinformatics	Piano Performance	Male	Yes	3.96	3.92	Cook_Christopher.jpg	No	
6	Davidson	Steve	example@internet.com	801-555-1234	Math		Male	Yes	3.65	3.28	Davidson_Steve.jpg	No	
7	Denny	David	example@internet.com	801-555-5678	Math and Economics	Computer Science	Male	Yes	3.55	3.70	Denny_David.jpg	No	
8	Fisher	John	example@internet.com	801-555-1234	Statistics	Math	Male	Yes	3.88	4.00	Fisher_John.jpg	Yes	
9	Frank	Owen	example@internet.com	801-555-5678	Statistics	Math	Male	Yes	3.99	3.98	Frank_Owen.jpg	No	
10	Hedges	Ally	example@internet.com	801-555-1234	Math	English	Female	Yes	3.95	3.91	Hedges_Ally.jpg	No	
11	Henderson	Jeff	example@internet.com	801-555-5678	Math	Computer Science	Male	Yes	4.00	4.00	Henderson_Jeff.jpg	No	
12	Hodgson	Gary	example@internet.com	801-555-1234	Statistics	Math and Japanese	Male	Yes	3.73	3.28	Hodgson_Gary.jpg	Yes	
13	Kohn	Taylor	example@internet.com	801-555-5678	Math		Male	Yes	3.95	3.90	Kohn_Taylor.jpg	No	
14	Long	Brittany	example@internet.com	801-555-1234	Mathematics		Female	Yes	3.26	3.57	Long_Brittany.jpg	Yes	
15	Part	David	example@internet.com	801-555-1234	Math	Molecular Biology	Male	Yes	3.78	3.20	Part_David.jpg	No	
16	Perkins	Owen	example@internet.com	801-555-5678	Math and Physics		Male	Yes	3.89	4.00	Perkins_Owen.jpg	No	
17	Rock	Aaron	example@internet.com	801-555-1234	Math		Male	Yes	3.69	3.42	Rock_Aaron.jpg	No	
18	Sherchan	Monique	example@internet.com	801-555-5678	Actuarial Science	Math	Female	Yes	3.36	3.69	Sherchan_Monique.jpg	No	
19	Shipp	Cade	example@internet.com	801-555-1234	Math and Economics		Male	Yes	3.93	4.00	Shipp_Cade.jpg	No	
20	Simkhada	Perry	example@internet.com	801-555-5678	Math		Male	Yes	3.70	3.74	Simkhada_Perry.jpg	No	
21	Smith	Jared	example@internet.com	801-555-1234	Math and Physics		Male	Yes	3.98	4.00	Smith_Jared.jpg	Yes	
22	Spencer	Mary	example@internet.com	801-555-5678	Math		Female	Yes	3.91	3.91	Spencer_Mary.jpg	No	
23	Tutfield	George	example@internet.com	801-555-1234	Math	Humanities	Male	Yes	3.80	3.58	Tutfield_George.jpg	Yes	
24	Willey	David	example@internet.com	801-555-5678	Neuroscience	Math	Male	Yes	3.95	3.94	Willey_David.jpg	Yes	
25	Young	Rob	example@internet.com	801-555-1234	Mechanical Engineering	Math	Male	Yes	3.56	3.89	Young_Rob.jpg	No	
26	Messick	Rachel	example@internet.com	801-555-6789	Statistics	Mathematics	Female	Yes	3.94	4.00	Messick_Rachel.jpg	No	



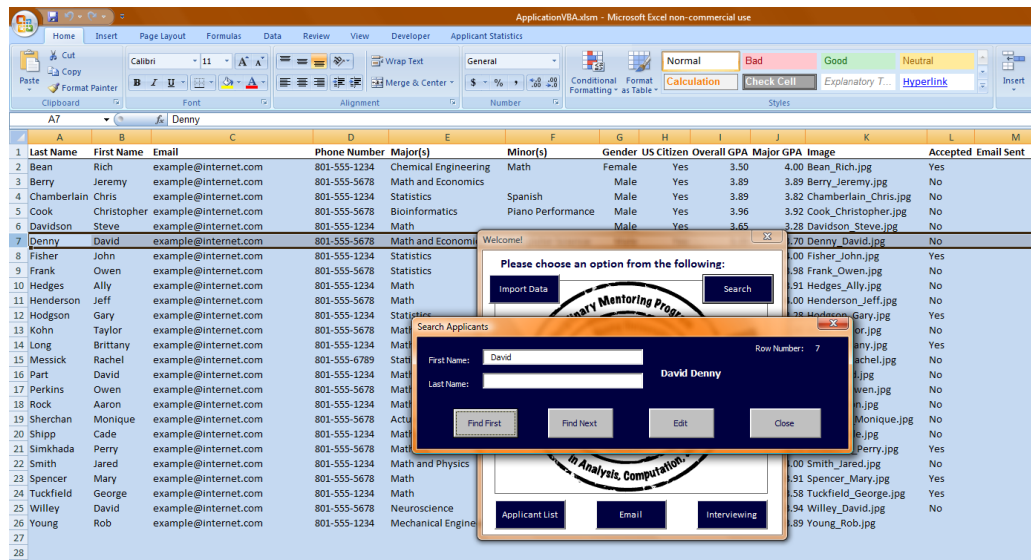
Once the data is placed into the worksheet, the temporary sheet is deleted and all the data in the worksheet is sorted by last name and is formatted correctly through the assistance of a recorded macro.



	A	B	C	D	E	F	G	H	I	J	K	L	M
	Last Name	First Name	Email	Phone Number	Major(s)	Minor(s)	Gender	US Citizen	Overall GPA	Major GPA	Image	Accepted	Email Sent
2	Bean	Rich	example@internet.com	801-555-1234	Chemical Engineering	Math	Female	Yes	3.50	4.00	Bean_Rich.jpg	Yes	
3	Berry	Jeremy	example@internet.com	801-555-5678	Math and Economics	Math	Male	Yes	3.89	3.89	Berry_Jeremy.jpg	No	
4	Chamberlain	Chris	example@internet.com	801-555-1234	Statistics	Spanish	Male	Yes	3.89	3.82	Chamberlain_Chris.jpg	No	
5	Cook	Christopher	example@internet.com	801-555-5678	Bioinformatics	Piano Performance	Male	Yes	3.96	3.92	Cook_Christopher.jpg	No	
6	Davidson	Steve	example@internet.com	801-555-1234	Math		Male	Yes	3.65	3.28	Davidson_Steve.jpg	No	
7	Denny	David	example@internet.com	801-555-5678	Math and Economics	Computer Science	Male	Yes	3.35	3.70	Denny_David.jpg	No	
8	Fisher	John	example@internet.com	801-555-1234	Statistics	Math	Male	Yes	3.88	4.00	Fisher_John.jpg	Yes	
9	Frank	Owen	example@internet.com	801-555-5678	Statistics	Math	Male	Yes	3.99	3.98	Frank_Owen.jpg	No	
10	Hedges	Ally	example@internet.com	801-555-1234	Math	English	Female	Yes	3.95	3.91	Hedges_Ally.jpg	No	
11	Henderson	Jeff	example@internet.com	801-555-5678	Math	Computer Science	Male	Yes	4.00	4.00	Henderson_Jeff.jpg	No	
12	Hodgson	Gary	example@internet.com	801-555-1234	Statistics	Math and Japanese	Male	Yes	3.73	3.28	Hodgson_Gary.jpg	Yes	
13	Kohn	Taylor	example@internet.com	801-555-5678	Math		Male	Yes	3.95	3.90	Kohn_Taylor.jpg	No	
14	Long	Brittany	example@internet.com	801-555-1234	Mathematics		Female	Yes	3.26	3.57	Long_Brittany.jpg	Yes	
15	Messick	Rachel	example@internet.com	801-555-6789	Statistics	Mathematics	Female	Yes	3.94	4.00	Messick_Rachel.jpg	No	
16	Part	David	example@internet.com	801-555-1234	Math	Molecular Biology	Male	Yes	3.78	3.20	Part_David.jpg	No	
17	Perkins	Owen	example@internet.com	801-555-5678	Math and Physics		Male	Yes	3.89	4.00	Perkins_Owen.jpg	No	
18	Rock	Aaron	example@internet.com	801-555-1234	Math		Male	Yes	3.69	3.42	Rock_Aaron.jpg	No	
19	Sherchan	Monique	example@internet.com	801-555-5678	Actuarial Science	Math	Female	Yes	3.36	3.69	Sherchan_Monique.jpg	No	
20	Shipp	Cade	example@internet.com	801-555-1234	Math and Economics		Male	Yes	3.93	4.00	Shipp_Cade.jpg	No	
21	Simkhada	Perry	example@internet.com	801-555-5678	Math		Male	Yes	3.70	3.74	Simkhada_Perry.jpg	Yes	
22	Smith	Jared	example@internet.com	801-555-1234	Math and Physics		Male	Yes	3.98	4.00	Smith_Jared.jpg	No	
23	Spencer	Mary	example@internet.com	801-555-5678	Math		Female	Yes	3.91	3.91	Spencer_Mary.jpg	Yes	
24	Tuckfield	George	example@internet.com	801-555-1234	Math	Humanities	Male	Yes	3.80	3.58	Tuckfield_George.jpg	Yes	
25	Viertel	Riley	example@internet.com	801-555-1234	Bioinformatics	Chemistry	Male	Yes	3.98	3.96	Viertel_Riley.jpg	Yes	
26	Willey	David	example@internet.com	801-555-5678	Neuroscience	Math	Male	Yes	3.95	3.94	Willey_David.jpg	Yes	
27	Young	Rob	example@internet.com	801-555-1234	Mechanical Engineering	Math	Male	Yes	3.56	3.89	Young_Rob.jpg	No	

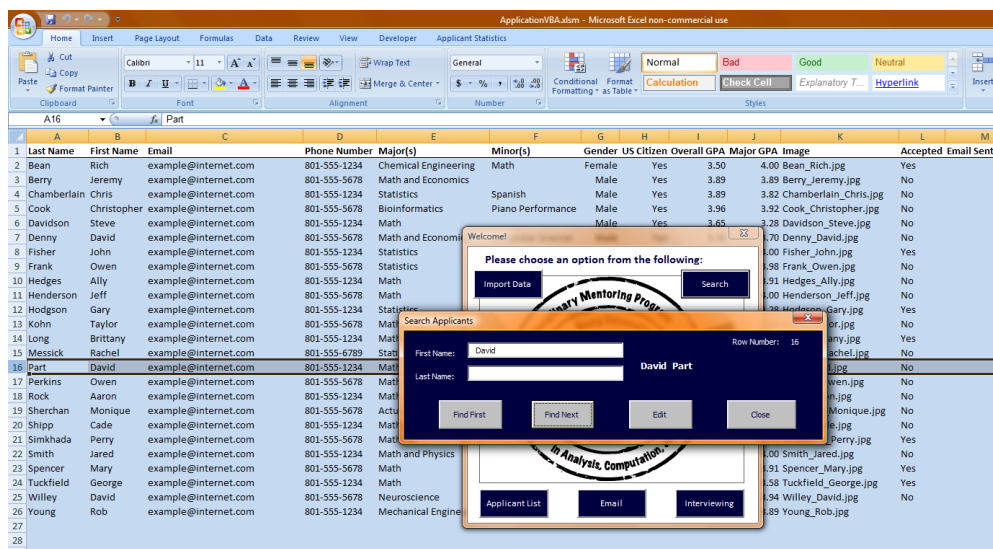
## Search

The purpose of the search button is to provide the ability for the user to search through the applicants by typing all or part of the applicant's first and/or last name. Once the required input has been entered in, the user is able to select either "Find First" or "Find Next." These functions employ the use of loops, if, and instring functions. If the "Find First" button is selected, the subprocedure will find the first instance of the inputted criteria in the "Applicant Info" worksheet. For example, if the first name of David was entered in the user form and "Find First" was selected, the name of the applicant David Denny would be returned.

	A	B	C	D	E	F	G	H	I	J	K	L	M
	Last Name	First Name	Email	Phone Number	Major(s)	Minor(s)	Gender	US Citizen	Overall GPA	Major GPA	Image	Accepted	Email Sent
2	Bean	Rich	example@internet.com	801-555-1234	Chemical Engineering	Math	Female	Yes	3.50	4.00	Bean_Rich.jpg	Yes	
3	Berry	Jeremy	example@internet.com	801-555-5678	Math and Economics	Math	Male	Yes	3.89	3.89	Berry_Jeremy.jpg	No	
4	Chamberlain	Chris	example@internet.com	801-555-1234	Statistics	Spanish	Male	Yes	3.89	3.82	Chamberlain_Chris.jpg	No	
5	Cook	Christopher	example@internet.com	801-555-5678	Bioinformatics	Piano Performance	Male	Yes	3.96	3.92	Cook_Christopher.jpg	No	
6	Davidson	Steve	example@internet.com	801-555-1234	Math		Male	Yes	3.65	3.28	Davidson_Steve.jpg	No	
7	Denny	David	example@internet.com	801-555-5678	Math and Economics						Denny_David.jpg	No	
8	Fisher	John	example@internet.com	801-555-1234	Statistics						Fisher_John.jpg	Yes	
9	Frank	Owen	example@internet.com	801-555-5678	Statistics						Frank_Owen.jpg	No	
10	Hedges	Ally	example@internet.com	801-555-1234	Math						Hedges_Ally.jpg	No	
11	Henderson	Jeff	example@internet.com	801-555-5678	Math						Henderson_Jeff.jpg	No	
12	Hodgson	Gary	example@internet.com	801-555-1234	Statistics						Hodgson_Gary.jpg	Yes	
13	Kohn	Taylor	example@internet.com	801-555-5678	Math						Kohn_Taylor.jpg	No	
14	Long	Brittany	example@internet.com	801-555-1234	Mathematics						Long_Brittany.jpg	Yes	
15	Messick	Rachel	example@internet.com	801-555-6789	Statistics	Mathematics	Female	Yes	3.94	4.00	Messick_Rachel.jpg	No	
16	Part	David	example@internet.com	801-555-1234	Math	Molecular Biology	Male	Yes	3.78	3.20	Part_David.jpg	No	
17	Perkins	Owen	example@internet.com	801-555-5678	Math and Physics		Male	Yes	3.89	4.00	Perkins_Owen.jpg	No	
18	Rock	Aaron	example@internet.com	801-555-1234	Math		Male	Yes	3.69	3.42	Rock_Aaron.jpg	No	
19	Sherchan	Monique	example@internet.com	801-555-5678	Actuarial Science	Math	Female	Yes	3.36	3.69	Sherchan_Monique.jpg	No	
20	Shipp	Cade	example@internet.com	801-555-1234	Math and Economics		Male	Yes	3.93	4.00	Shipp_Cade.jpg	No	
21	Simkhada	Perry	example@internet.com	801-555-5678	Math		Male	Yes	3.70	3.74	Simkhada_Perry.jpg	Yes	
22	Smith	Jared	example@internet.com	801-555-1234	Math and Physics		Male	Yes	3.98	4.00	Smith_Jared.jpg	No	
23	Spencer	Mary	example@internet.com	801-555-5678	Math		Female	Yes	3.91	3.91	Spencer_Mary.jpg	Yes	
24	Tuckfield	George	example@internet.com	801-555-1234	Math	Humanities	Male	Yes	3.80	3.58	Tuckfield_George.jpg	Yes	
25	Viertel	Riley	example@internet.com	801-555-1234	Bioinformatics	Chemistry	Male	Yes	3.98	3.96	Viertel_Riley.jpg	Yes	
26	Willey	David	example@internet.com	801-555-5678	Neuroscience	Math	Male	Yes	3.95	3.94	Willey_David.jpg	Yes	
27	Young	Rob	example@internet.com	801-555-1234	Mechanical Engineering	Math	Male	Yes	3.56	3.89	Young_Rob.jpg	No	

If the “Find Next” button is clicked, the subprocedure will find the next instance of the inputted criteria in the “Applicant Info” worksheet. With this example, the “Find Next” button would return the name of David Part. If clicked again, the applicant David Willey would be displayed.



Although the search function is pretty useful, it becomes even more useful when you can find an applicant and edit any applicant data that needs to be changed. For example, the applicant may inform the user of an email address or phone number change, or his/her GPA could have adjusted. The “Edit” button on this user form allows for the user to edit the applicant data. The edit form is able to open the applicant’s information by linking the row number from the search form to the row number for the edit form.

```
Private Sub cmdEdit_Click()
'After selecting an applicant to edit, opens the edit form.
If lblNameFound = "" Then
MsgBox "You must find a record to edit before opening the Edit form."
Else
frmEditData.Show
rowNum = frmSearch.lblRowNum.Caption
End If
End Sub
```

The screenshot shows a form titled 'Edit Applicant Data'. It contains the following fields and values:

- Applicant Name: David Part
- Email: example@internet.com
- Phone Number: 801-555-1234
- Major(s): Math
- Minor(s): Molecular Biology
- Gender: Male (selected from a dropdown)
- US Citizen: Yes (selected with a radio button)
- Overall GPA: 3.78
- Major GPA: 3.20

At the bottom of the form, there are four buttons: 'Previous', 'Next', 'Save', and 'Close'. To the right of the form is a small image of a smiling boy giving a thumbs up.

The applicant name, email, phone number, major, minor, overall GPA, and major GPA can be edited simply by deleting the current data and typing in the new, correct information. If an applicant has mistakenly put the wrong gender on his/her application, a change can be made by selecting the correct gender in the combo box. Finally, a change in the U.S. citizenship of an applicant is changed by selecting either the “Yes” or “No” option button.

Once the changes have been made, the user simply clicks the “Save” button and those changes will be saved by writing the information to the “Applicant Info” worksheet.

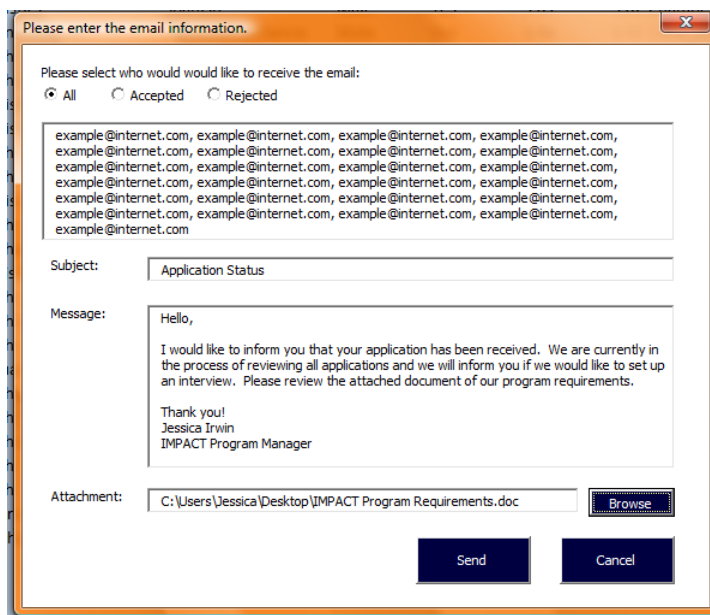
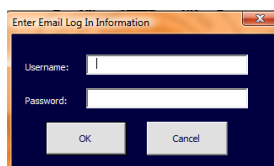
## Applicant List

By selecting the “Applicant List” option, the user is able to view each applicant’s information. Their name, information, and picture are brought in from the worksheet and displayed on the form. The user is able to click the “Previous” and “Next” buttons to see each applicant, which is done simply by “if” functions. Just like the search form, this form allows the user to edit the applicant’s information. Once “Edit” is clicked, the same user form used to edit the applicant data is displayed and the user is able to edit the data. This is done the same way as the search form.



## Email

A very useful function of this program is the ability for the user to email applicants. Once the “Email” option has been selected, an input box is displayed. The user is to enter the username and password of his/her email address. The inputs are then saved as a string to be used to send the email. Since we use a Gmail account in the IMPACT Program, this function was created to work with Gmail. After the user has entered the username and password and clicked OK, another user form is displayed:



Here, the user is able to email all applicants, accepted applicants, or rejected applicants by selecting one of the three option buttons. Once a option button has been selected, the email addresses appear in the box below the buttons. In order to populate the text box with the correct emails, I used a “Do Until” loop that loops through all the email addresses on the “Applicant Info” worksheet. If the “All” button is selected, the loop simply brings in all the emails until it comes to a row with nothing in it. If the “Accepted” button is select, the loop brings in the emails that have the word “Yes” under the “Accepted” column. This is done by a



simple “If” function. The same process is used for the “Rejected” button. Once all of the emails have been inputted, the last two characters of the text box are deleted (“,” “”).

```
Private Sub optAll_Click()
'Populates all the recipients of the email message on the Email form.
Dim r As Integer
txtRecipients.Text = ""

r = 2
Do Until Sheets("Applicant Info").Cells(r, 3).Value = ""
    txtRecipients.Text = Sheets("Applicant Info").Cells(r, 3).Value & ", " & txtRecipients.Text
    r = r + 1
Loop

txtRecipients.Text = Left(txtRecipients.Text, Len(txtRecipients.Text) - 2)
End Sub

Private Sub optAccepted_Click()
'Populates the accepted recipients of the email message on the Email form.
Dim r As Integer
txtRecipients.Text = ""

r = 2
Do Until Sheets("Applicant Info").Cells(r, 12).Value = ""
    If Sheets("Applicant Info").Cells(r, 12).Value = "Yes" Then
        txtRecipients.Text = Sheets("Applicant Info").Cells(r, 3).Value & ", " & txtRecipients.Text
    End If
    r = r + 1
Loop

txtRecipients.Text = Left(txtRecipients.Text, Len(txtRecipients.Text) - 2)
End Sub
```

The user then can type a subject, message, and attach a file to the email. Once the email is ready to be delivered, the “Send” button sends the email to the selected applicants. The “sendGmail” function provides the means to send the email. The username and password from the “Log In” user form are passed to the function, along with the message, subject, and attachment that are typed by the user in the “Email” user form. The recipients of the email depend on which option button was checked, as explained before. With the “All” option button selected, all the email addresses are passed to the function with a “Do Until” loop. With the “Accepted” or “Rejected” option button selected, it gets a bit more complicated. Through the use of a split function, each of the email addresses in the text box of recipients was split into an array, called “splitEmails.” Each email address became a value in the array and with the help of a “For” loop, each email address in the array was passed to the “sendGmail” function.

If the email was successfully sent to an applicant, the date and time it was sent appears in the “Email Sent” column. If the email failed, the text “Bad Email” appears in the “Email Sent” column of the applicant.

```
r = 2

'If want to email all applicants, the recipients are all the emails on the applicant info tab.
'Shows when the email was sent if successful, and if failed shows it was a bad email.
If optAll.Value = -1 Then
    Do Until Sheets("Applicant Info").Cells(r, 3).Value = ""
        If sendGmail(frmLogin.txtUsername.Text, frmLogin.txtPassword.Text, message, Sheets("Applicant Info").Cells(r, 3).Value, subject, attachment) Then
            Sheets("Applicant Info").Cells(r, 13).Value = Now
        Else
            Sheets("Applicant Info").Cells(r, 13).Value = "Bad Email"
        End If
        r = r + 1
    Loop
End If

'If want to email accepted applicants, the recipients are the accepted (or "Yes" in the 12th column) applicants on the applicant info tab.
'Shows when the email was sent if successful, and if failed shows it was a bad email.
If optAccepted.Value = -1 Then
    splitEmails = Split(txtRecipients.Text, ", ")
    For x = 0 To UBound(splitEmails)
        If sendGmail(frmLogin.txtUsername.Text, frmLogin.txtPassword.Text, message, splitEmails(x), subject, attachment) Then
            Do Until Sheets("Applicant Info").Cells(r, 3).Value = ""
                If Sheets("Applicant Info").Cells(r, 12).Value = "Yes" Then
                    Sheets("Applicant Info").Cells(r, 13).Value = Now
                End If
                r = r + 1
            Loop
        Else
            Sheets("Applicant Info").Cells(r, 13).Value = "Bad Email"
        End If
    Next
End If
```

## Interviewing

The Interviewing button allows a user to select an interview time for each applicant. A list box was used to load the day and time of each interview session from the “Interview Schedule” worksheet.

```
Private Sub UserForm_Initialize()
    Sheets("Interview Schedule").Activate
    rowNum = 2
    showData

    'Populate the list box.
    r = 2
    x = 0
    Do Until Sheets("Interview Schedule").Cells(r, 1).Value = ""
        If Sheets("Interview Schedule").Cells(r, 3).Value = "" Then
            lstInterview.AddItem Sheets("Interview Schedule").Cells(r, 1).Value
            lstInterview.List(x, 1) = Sheets("Interview Schedule").Cells(r, 2).Text
            x = x + 1
        End If
        r = r + 1
    Loop
End Sub
```

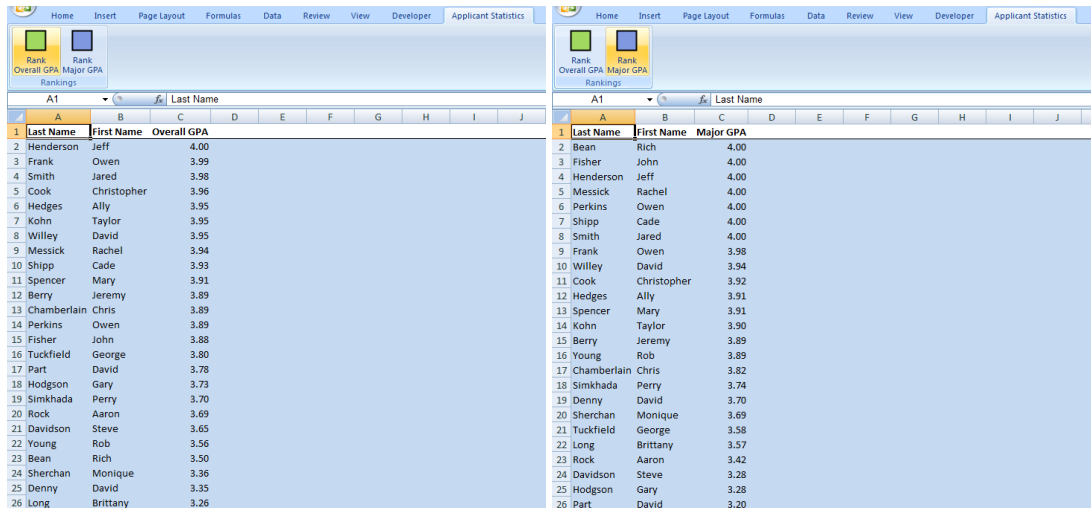
The user is able to scroll through the each applicant using the “Previous” and “Next” buttons. Once the desired applicant is displayed, a day and time for the interview can be selected from the list and if the user clicks “Save”, the interview day and time will be listed next to the applicant’s name on the “Interview Schedule” worksheet. The string variable “listTime” was created by combining the selected row’s value in the first column (the day of the week) and the value of the second column (the time). Then, a loop is created to match the “listTime” value to the “time” string, which is the day and time from the “Interview Schedule” worksheet. Once these two values are equal, the subprocedure writes the name of the applicant next to that day and time. Then day and time that was assigned to the applicant disappears from the list in order to prevent double-scheduling interviews.

```
Private Sub cmdSave_Click()
    Dim time As String
    'Write the time selected in the user form to the worksheet and delete the time because it is no longer available.
    r = 2
    y = lstInterview.ListIndex
    listTime = lstInterview.List(y, 0) & " " & lstInterview.List(y, 1)
    Do Until Sheets("Interview Schedule").Cells(r, 1).Value = ""
        time = Sheets("Interview Schedule").Cells(r, 1).Value & " " & Sheets("Interview Schedule").Cells(r, 2).Text
        If listTime = time Then
            Sheets("Interview Schedule").Cells(r, 3).Value = lblName.Caption
            lstInterview.RemoveItem lstInterview.ListIndex
            Exit Sub
        End If
        r = r + 1
    Loop
End Sub
```



## Applicant Statistics Ribbon Tab

The “Applicant Statistics” ribbon tab consists of two buttons: (1) Rank Overall GPA and (2) Rank Major GPA.



Rank	Last Name	First Name	Overall GPA
1	Henderson	Jeff	4.00
2	Frank	Owen	3.99
3	Smith	Jared	3.98
4	Cook	Christopher	3.96
5	Hedges	Ally	3.95
6	Kohn	Taylor	3.95
7	Willey	David	3.95
8	Messick	Rachel	3.94
9	Shipp	Cade	3.93
10	Spencer	Mary	3.91
11	Berry	Jeremy	3.89
12	Chamberlain	Chris	3.89
13	Perkins	Owen	3.89
14	Fisher	John	3.88
15	Tuckfield	George	3.80
16	Part	David	3.78
17	Hodgson	Gary	3.73
18	Simkhada	Perry	3.70
19	Rock	Aaron	3.69
20	Davidson	Steve	3.65
21	Young	Rob	3.56
22	Bean	Rich	3.50
23	Sherchan	Monique	3.36
24	Denny	David	3.35
25	Long	Brittany	3.26

Rank	Last Name	First Name	Major GPA
1	Bean	Rich	4.00
2	Fisher	John	4.00
3	Henderson	Jeff	4.00
4	Messick	Rachel	4.00
5	Perkins	Owen	4.00
6	Shipp	Cade	4.00
7	Smith	Jared	4.00
8	Frank	Owen	3.98
9	Willey	David	3.94
10	Cook	Christopher	3.92
11	Hedges	Ally	3.91
12	Spencer	Mary	3.91
13	Kohn	Taylor	3.90
14	Berry	Jeremy	3.89
15	Young	Rob	3.89
16	Chamberlain	Chris	3.82
17	Simkhada	Perry	3.74
18	Denny	David	3.70
19	Sherchan	Monique	3.69
20	Tuckfield	George	3.58
21	Long	Brittany	3.57
22	Rock	Aaron	3.42
23	Davidson	Steve	3.28
24	Hodgson	Gary	3.28
25	Part	David	3.20

These buttons allow the user to rank the overall and major GPA of the applicants in order from highest to lowest. The subprocedure was started by creating a macro, with some additional VBA modifications. First, information from the “Applicant Info” worksheet is copied onto the “Overall GPA Ranking” or “Major GPA Ranking” worksheet. The information is then sorted by either overall or major GPA. Lastly, the worksheet and data are reformatted (excess columns deleted, autofit columns, reformat numbers, etc.). This provides the user with some valuable comparative information that can prove to be useful in his/her decision making.

```
Sub overallGPAranking(control As IRibbonControl)

Application.ScreenUpdating = False

'Copy items on Applicant Info tab.
Sheets("Applicant Info").Select
Cells.Select
Selection.Copy

'Paste items on Overall GPA Ranking tab.
Sheets("Overall GPA Ranking").Select
Range("A1").Select
Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks:=False, Transpose:=False
Application.CutCopyMode = False

'Sort by Overall GPA from largest to smallest.
ActiveWorkbook.Worksheets("Overall GPA Ranking").Sort.SortFields.Clear
ActiveWorkbook.Worksheets("Overall GPA Ranking").Sort.SortFields.Add Key:=Range("I2:I5000"), _
    SortOn:=xlSortOnValues, Order:=xlDescending, DataOption:= _
    xlSortTextAsNumbers

With ActiveWorkbook.Worksheets("Overall GPA Ranking").Sort
    .SetRange Range(Range("A1"), Range("A1").End(xlDown).End(xlToRight))
    .Header = xlYes
    .MatchCase = False
    .Orientation = xlTopToBottom
    .SortMethod = xlPinYin
    .Apply
End With

'Delete excess information.
Columns("C:H").Select
Selection.Delete Shift:=xlToLeft

Columns("D:G").Select
Selection.Delete Shift:=xlToLeft
```

## Learning and Conceptual Difficulties

### Importing and Parsing Data from Word Document

One of the biggest difficulties for me in completing my project was learning how to import and parse the data from a Word document. Even though I had not initially proposed to do this in my project, I decided that this method of inputting data would be a lot more user friendly and more useful than inputting the data by hand into a user form. I spent a large portion of my time searching Google and learning how to do this. I found that it is more common for VBA programmers to write data from Excel into Word, instead of the other way around, which is what I was trying to accomplish. I found quite a few forums and threads of people attempting to write data from Word to Excel, but very few that I tried actually worked. Then I finally found a site that gave me the foundation and I was able to build my subroutine off of that example.

I also had to spend a large amount of my time parsing the data from the Word document. The IMPACT Program application, when saved as a string in Excel, had a lot of symbols and extra spaces, which provided me with some great practice on “stepping into” the subroutine to figure out exactly where the values were. There were many times where I was trying to use the `instr` and `mid` functions and I had to just try random numbers to find the data I wanted to extract out and write into the worksheet. I was able to successfully import and parse the data, but not without a lot of time and effort!

### Imported Data Formatting Issues

As the imported data from Word was brought into Excel as a string, I was presented with the problem of reformatting the GPA numbers. These numbers were formatted as values and not as numbers, like I needed them to be. This affected my Overall and Major GPA Rankings buttons—the subroutine did not recognize the imported GPAs as values and left them out. To fix this, I was able to find the `Val` function. Basically, the `Val` function accepts a string as input and returns the numbers found in that string. For example, the `Val` function with the string “I would like 10 cookies” would return the number 10. With this function I was able to change GPAs from values into numbers.

### Interview Scheduling User Form

The interview scheduling user form took me some time to figure out. I felt that I wasn’t too familiar with how list boxes worked. As can be expected, the part of the form that took me the longest to complete because of the conceptual difficulty was the “Save” function. I was able to ask and get assistance from some fellow classmates, which helped out a lot. In the end, I think that I was successful with the user form and I am pleased with how it turned out.

### No Simultaneous Editing and Viewing Feature

One negative aspect of doing the application selection in the program I have written is that it does not allow for simultaneous editing and viewing, which was why I have used a GoogleDoc for this process for the past few years. Although it isn’t a big problem, each time there is a change I will have to update and email the most recent versions to our program directors.