**Project Report**

Anastasia Zelenova i6205531

Alexander Yalunin i6205532

Tutor: Janos Flesch

Table of Content

1. **Introduction**…………………………………………………………………………..3
2. **User guide**……………………………………………………………………………..4
3. **Summary of HomeSelector Variables and Calculations**……………………….…10
   1. Description of variables …………………………………………………..….10
   2. Description of calculations…………………………………………………...12
4. **Manager`s guide**…………………………………………………………………….14
   1. Changing Spreadsheet Data…………………………………………………..14
   2. Changing Visual Basic Program……………………………………………...15
5. **Limitations**…………………………………………………………………………..18
6. **Conclusion**…………………………………………………………………………...19
7. **References**…………………………………………………………………………...20
8. **Introduction**

The real estate market in Moscow is a vast, fast-growing and complex market. Thousands of different apartments, studios, and houses with dozens of variables that need to be considered, make it almost impossible for the buyer to make the optimal choice. Since the market is so large and complex, buyers, first, spend a lot of time trying to find something convenient, second, they lose focus on details trying to process a large amount of information, third, physically cannot view all sales and, finally, cannot assess the usefulness of each offer for them. To solve these problems and help the buyer make the right choice, we have developed an information system that will allow users to find the perfect accommodation based on their preferences, also known as Decision Support System (DSS).

Decision Support System is a computerized information system used to facilitate quality business decision-making for an organization or a business. A DSS lets users analyze a variety of data from many sources, and extract information that can be used to solve problems and make better decisions. [1]

This kind of application will have a wide target audience in the real estate market, including people trying to find housing (e.g. families, first home buyers, investors).

The user can configure up to 29 variables in HomeSelector decision support system, including, for example, price range, city area and distance to the supermarket. In addition, he will have the opportunity to establish levels of importance for all of them. Taking all of this into account and putting this information into a carefully designed scoring function, we ensure that our DSS will recommend the optimal accommodation for the user.

1. **User guide**

Having launched the program, the user sees *the Start page* of the application and can go to the main part of it by clicking on the *Start* button. *The Main page* consists of a set of questions divided into 6 logical groups, namely: *Main characteristics, Square meters, Distances, Additional, Floors, and Map.* On the last tab*, Map*, a user is able to go to the next page, *the Summary page*, by clicking on the *Next* button. *The Summary page* represents an overview of all characteristics selected by a user and provides an opportunity to set an importance level for every logical group. Then the user can either go back and change any characteristics (by clicking on the *Back* button or *Change* buttons) or continue and go to the next page, *the Result page*, by clicking on the *Show results* button. *The Result page* is composed of the list box, reflecting adequate apartments sorted by descending of a total score,the *Show top 10/3 button* which allows showing either top 10 or top 3 apartments and the *Back* button thatis necessary to go back to *the Result page*. A user can double click on an apartment and go to the next page, *the Description page*, which represents a whole description of a selected apartment. Further, the user has several possible ways: to close the application by clicking on the *Exit* button or go back to the *Result* page to look at other apartments in detail and return to *the Main page* and *the Summary page* by clicking on the *Back* buttons.

Having considered the full functionality of the program, let's move on to a more detailed description of all user capabilities step by steps.

1. Having launched the program, click on the *Start* button to go to *the Main page*.

The first tab on which the user gets after starting the program is *Main characteristics*. In addition, there are also other tabs such as *Square footage, Distances, Additional,* etc. A user can use these tabs to switch from one logical group of questions to another or use the *Next* and *Back* buttons. The first thing the user should specify is the price range for an apartment using two scroll bars which allow setting a minimum price that is less than or equal to the maximum price. If the price range is not important, then the user selects checkbox *“Irrelevant”*.

1. Set the price range by scrolling the slider or pressing the arrows on the scroll bars.

The next main characteristic is the number of rooms in an apartment. The user can select any number of rooms at the same time or not select it at all and select the checkbox *“Irrelevant”*.

1. Select the desired number of rooms in an apartment.

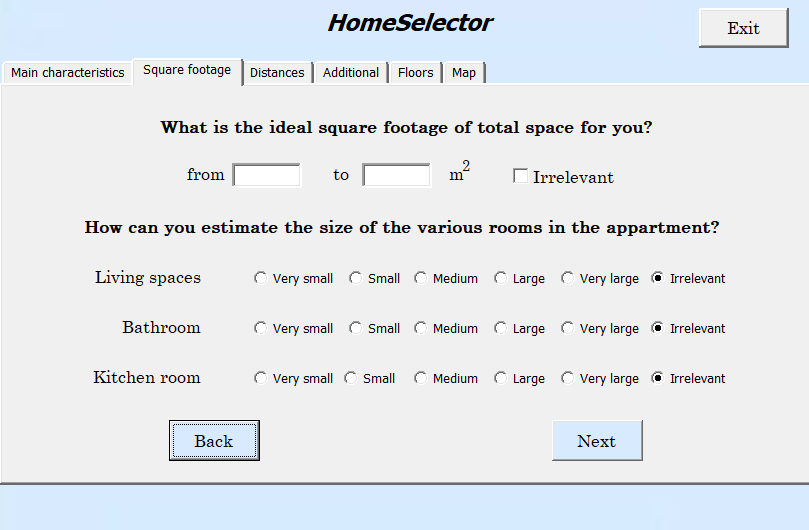
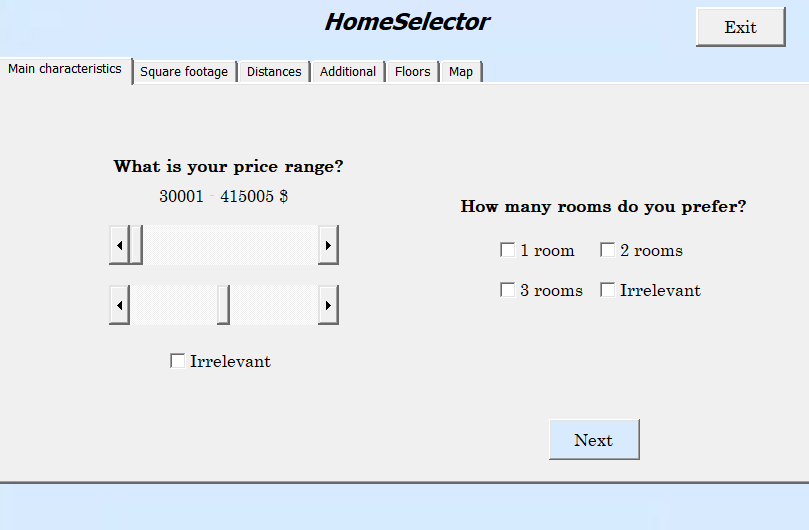


Figure 1: Screenshots of the Main characteristics page and Square footage page

1. Click on the *Next* button or *Square footage* tab to go to the next logical group of questions.

The first part of the page includes the choice of the total area of ​​the apartment. The user should specify the minimum and maximum square footage in the text boxes or select the checkbox *“Irrelevant”*. If the user sets an incorrect interval, then by clicking on the *Next* button a warning appears about the incorrectly specified interval.

1. Set the correct interval of the square footage of the total space of an apartment.

On the next part of the page, the user can select the size of the kitchen room, bathroom or living spaces using the option buttons: *Very small, Small, Medium, Large, Very large*. For user convenience, the default value is *“Irrelevant”*.

1. Choose the size of living spaces, bathroom and kitchen room.
2. Click on the *Next* button or *Distances* tab.

*The Distances page* contains two similar questions about the distance to the center of Moscow and a supermarket. The user can choose these distances by using scrollbars or can select the checkbox *“Irrelevant”*.

1. Specify the distance to center and supermarket by scrolling the slider or pressing the arrows on the scroll bars.

The next question on the Distances page is *“How close to a metro station do you want to live?”* including two combo boxes. The first combo box is designed to determine the amount of time the user would like to spend on the road to the metro station: *“Irrelevant”, up to 5, 10, 15, 30 or 60 minutes*. The second combo box indicates how the user prefers to get to the metro station: *“Irrelevant”,* *get on foot, get by transport*.

1. Select the distance in minutes and the way to get to the metro station.

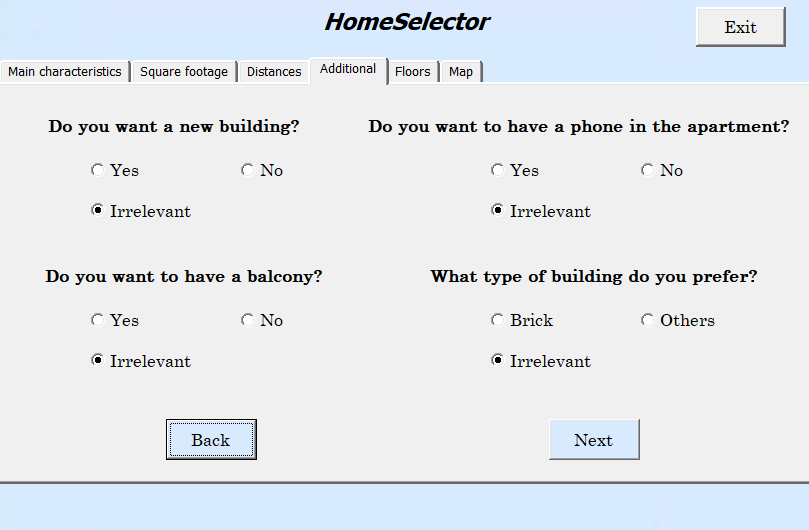
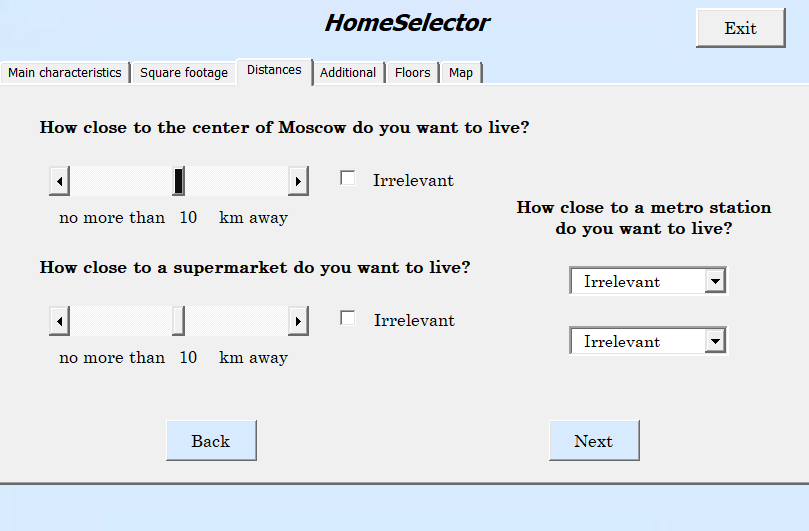


Figure 2: Screenshots of the Distances page and Additional page

1. Click on the *Next* button or *Additional* tab.

On *the Additional page*, the user can observe four questions regarding the telephone and the balcony in the house, the type and age (new or old) of the building. Variants of answers to questions are presented in the form of a choice of three possible options: *Yes, No, Irrelevant*. However, the answers to the question *"What type of building do you prefer?"* are presented in a different way: *Brick, Others, Irrelevant*. For user convenience, the default value is *“Irrelevant”*.

1. Indicate the required condition regarding the telephone and the balcony in the house, the type, and age of the building.
2. Click on the *Next* button or *Floors* tab.

On *the Floors page*, the user can select the number of floors in the building (*Irrelevant, from 5, 10, 15, 20, 25, 30 floors*) and specify the interval of the floors on which he or she would like to live by using combo boxes. For user convenience, the default value is *“Irrelevant”*.

1. Choose the number of floors in the building.

Besides, the user should select the minimum floor in the first combo box and the maximum floor in the second combo box. If the user selects the 10th floor in the first combo box and then tries to select the 2nd floor in the second combo box, the value in the first combo box becomes 2 and vice versa. This feature was added in order to avoid incorrect data entry and possible further errors in calculations, so the user should indicate the interval of floors properly. Moreover, the user can select the floor number only in the first combo box, while in the second combo box select the value *“Irrelevant”* and vice versa. This feature is useful when the user has preferences for only the minimum or maximum floor number.

1. Specify a range of floors.

In addition, if the user has any preferences about the top floor of the building, he or she can choose one of these options in the combo box: *Top floor is irrelevant, Only the top floor, Only not the top floor.* Also, there is a combo box *“Not the first floor”* if the user does not want to live on the first floor. For user convenience, the default value is *“Top floor is irrelevant”*.

1. Indicate preferences concerning the top floor and the first floor.

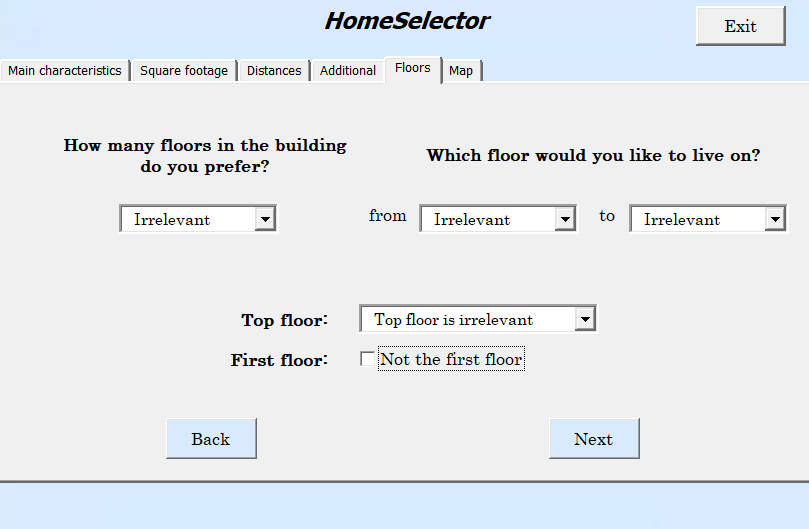
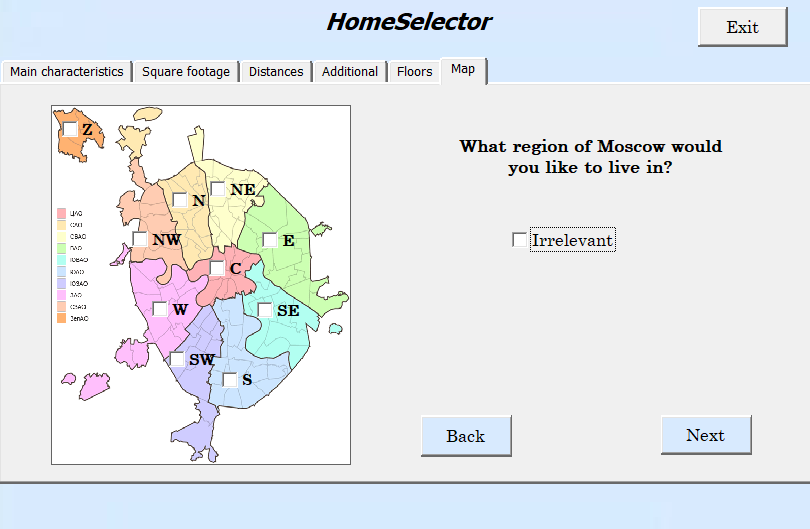
 

Figure 3: Screenshots of the Floors page and Map page

1. Click on the *Next* button or *Map* tab.

On *the Map page*, the user can observe a map of Moscow regions. Moscow is divided into ten districts formed for the administration of the respective territories. List of administrative districts of the city of Moscow: Central - C, Northern - N, Northeast - NE, East - E, Southeast - SE, South - S, Southwest - SW, West - W, Northwest - NW and Zelenograd - Z. To select the desired region, the user should check this region on the map. If the location of an apartment is not important, the user can select the checkbox *“Irrelevant”*.

1. Choose the desired regions of Moscow.
2. Click on the *Next* button.

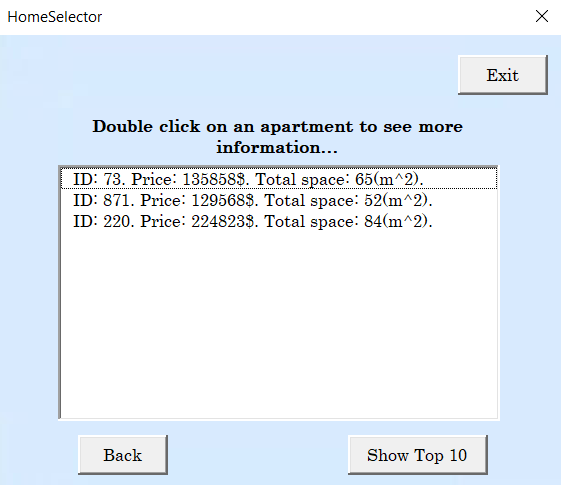
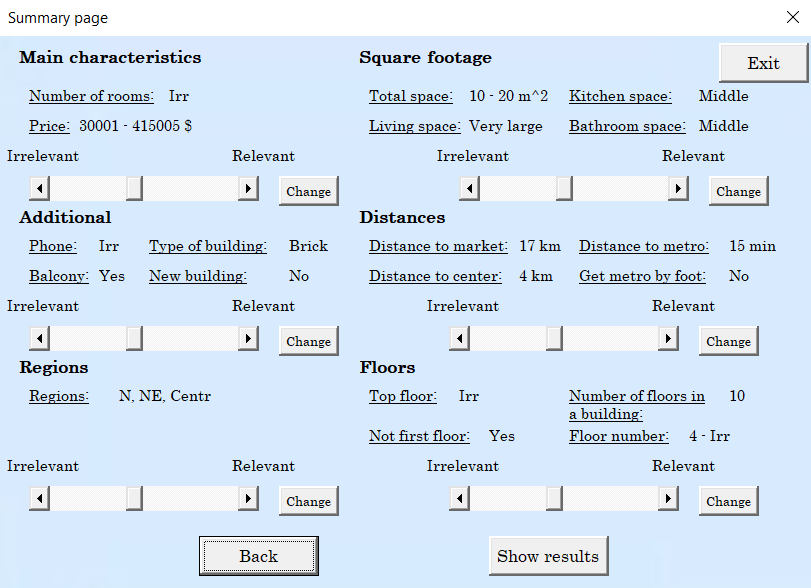


Figure 4: Screenshots of the Summary page and Result page

*The Summary page* provides the ability to check the characteristics selected by the user and return to *the Main page* to change some of them before receiving the results. All characteristics are divided, as on *the Main page*, and the user can increase the level of importance of each group by using scrollbars.

1. Check all conditions and go back by clicking the *Back* button *or Change* button if you need to change something.
2. Set the importance level for logical groups.
3. Click on the *Show results* button.

*The Result page* contains the list box of apartments that are most suitable for selected conditions. Each element is represented as a string consisting of id, price and total space of an apartment.

1. Double click on an apartment to see more details.

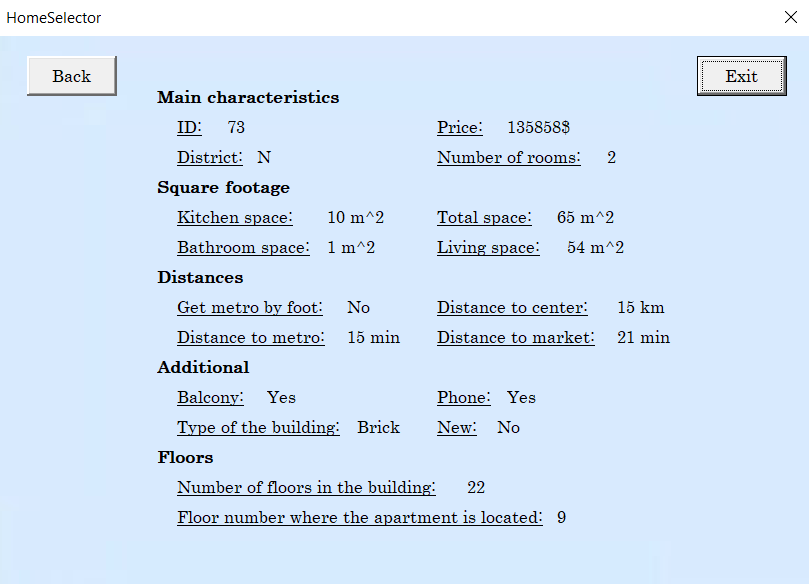


Figure 5: Screenshots of the Description page

On *the Description page*, the user can observe all information related to the apartment.

1. Click on the *Exit* button to close the application, or usethe *Back* buttons to return to *the Main page, the Summary page,* and *the Result page*.
2. **Summary of HomeSelector Variables and Calculations**
   1. Description of variables

The dataset we used to develop our DSS contains 1172 rows of real apartments sales in Moscow in 2006. It contains 34 columns which we used to create 29 variables and 6 importance levels. The variables are: n (apartment number); rooms (amount of rooms); price (the price of an apartment measured in dollars); totsp (total space measured in m^2); livesp (living space measured in m^2); kitsp (kitchen space measured in m^2); distcenter (distance to center in km); distmetro (distance to the metro station in minutes); onfoot (1 -- get metro on foot, 0 -- get metro by transport); brick (1 -- brick, monolith, 0 - other); phone (1 - there is a phone in an apartment, 0 - no); balcony (1 - there is at least one balcony or loggia, 0 - no); floorbetween (1 - the apartment is not on the first or top floor, 0 - the apartment is on the first or top floor); new (1 - primary market, 0 - secondary market); floors (number of floors in the building), nfloor (floor number where the apartment is located); firstfloor (1 -- the apartment is on the first floor); topfloor (1 -- the apartment is on the top floor); N, NE, E, SE, S, SW, W, NW, Centr, Zelen are dummy variables or indicators of the administrative district; elevator (1 - there is, 0 - no); extraspace (extra space measured in m^2); metro\_num (id metro station); timesupermarket (distance to a supermarket measured in minutes); distsupermarket (distance to a supermarket measured in km). Besides, the description of columns also presented in the Description sheet.

The UserData sheet contains all variables (features) that were used for calculations, grouped by meaning: Price range, Number of rooms, Room spaces, Distances, Floors information, Regions of the city, Additional information, Relevance levels.

Price range

Price range is stored in *min* and *max* variables on the UserData sheet.

Number of rooms

Our application provides a choice of apartments with 1, 2 and 3 rooms, so we store information about how many rooms the user has selected. For example, if a user chooses only 1 room apartment, in our database on the UserData sheet we will store the value “TRUE” for the number of rooms equal to *1*.

Rooms spaces

The value of total space selected by the user is stored as a minimum (*from* variable) and maximum (*to* variable) value, while the size of living, bathroom, and kitchen spaces measured from 1 to 5, where 1 - *Very small*, 2 - *Small*, 3 - *Medium*, 4 - *Large*, 5 - *Very large*.

Distances

There are three types of distances, namely, distance to metro (5, 10, 15, 30, 60 min), distance to the center of Moscow (from 0 to 20 km) and distance to the supermarket (from 0 to 20 km) that are stored as an integer values in UserData sheet in variables *metro minutes*, *center*, *supermarket*. Besides, the user can select a way how he or she wants to get to the metro station (*get by transport or get on foot*). If the user chooses *“On foot”*, we will store the value “TRUE” for this metric (*on foot* variable).

Floors information

Characteristics relating to the *Floors*, such as the number of floors in the building and the range of floors on which the user would like to live, are stored as numerical values in the variables: *in building* and *from/top*, relatively. And the information considering the first and last floor in the house is stored as a boolean value in the variables: *not first floor* and *top floor*, relatively.

Regions of the city

Due to the fact that the user can select several regions at once, dummy variables (*N, NE, E, SE, S, SW, W, NW, Centr, Zelen*) have been created for each region that takes boolean values.

Additional information

Each variable in this section (*new, telephone, balcony, brick*) is boolean since the questions were formulated as close-ended questions.

Relevance levels

A *Relevance level* variable is a variable that stores a value defined by the user that reflects the level of how important a logical group is to it. Due to the fact that HomeSelector has 6 logical groups, there are 6 *Relevance levels* measured from 1 to 10: *Main (price, rooms), Square* *footage, Distances, Additional, Floors, Map.*

* 1. Description of calculations

There are 3 levels of calculations. The first level computes the weights for each individual variable. The second level computes the weights for groups of variables. The third level computes the final score.

Firstly, we compute 29 weights for each of our variables. If the user chooses the value that is equal to the corresponding value of the example or if he chooses that this variable is irrelevant than the weight for this variable equal “TRUE”, otherwise it is “FALSE”. This binary principle is applicable to each of our variables except the *price*. The *price* is a continuous variable, its weight is equal to the value between 0 and 1 if the price of example is inside the user’s price range and 0 otherwise. The example of *price* calculation one can see on the *Example* sheet.

Secondly, we compute weights for groups of variables. The user sets the value between 0 and 10 of how important the corresponding group is. To make each group even, we weight them by the number of variables that are inside these groups. For example, the *Square* *footage* group contains 4 variables that represent different spaces, let's assume that 3 of them became “TRUE” and the user set the importance level to 8. Then, the score for this group will be: 8 \* (¾) = 6. However, we don’t weight the *Map* group, because if, say, the user chooses 3 regions out of 10, only one variable can be “TRUE”, because of the accommodation can’t be in more than one area of the city.

Finally, we compute the final score. Each of our variables except the price represents the *soft constraint*, i.e. they cannot affect each other. The *price* is a *hard constraint*: if its weight is zero then the entire score is zero. This way we will not show accommodations for the user that are out of his price range. If the price is non-zero, then the final score takes into account the groups weights and sums them up. After all, the scores for examples are sorted and those with the highest score are shown to the user.

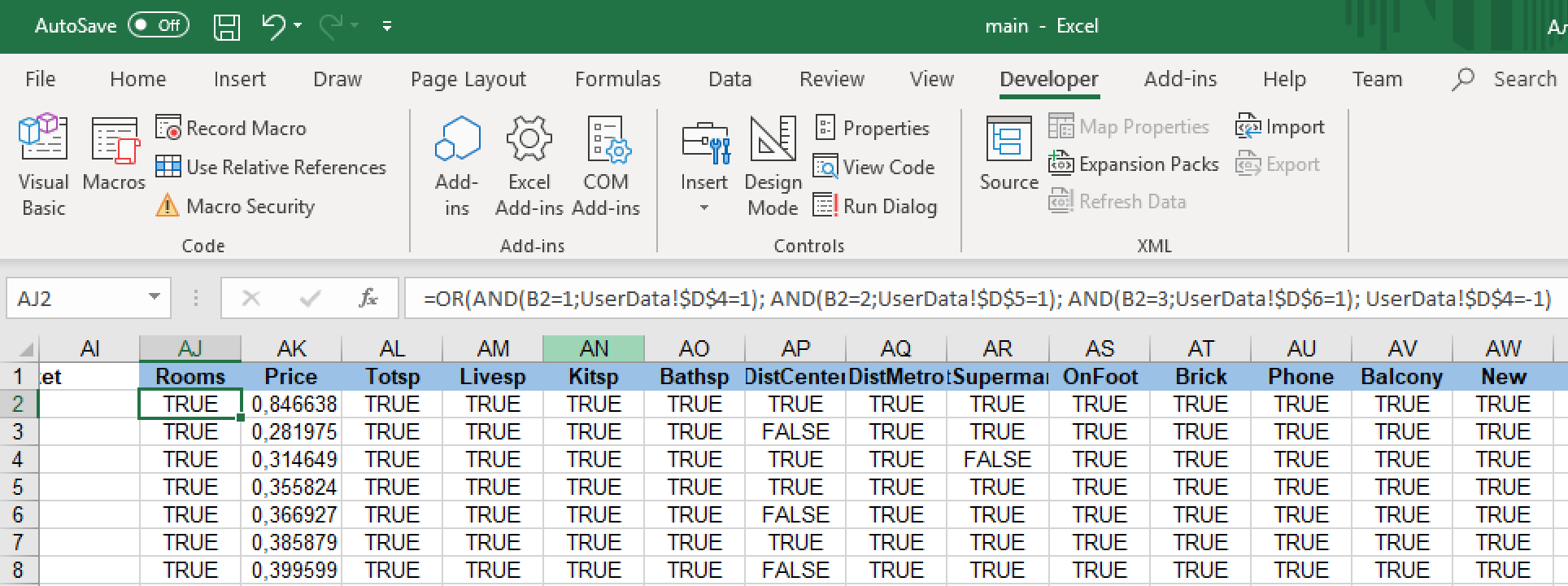
1. **Manager’s guide**
   1. Changing Spreadsheet Data

There are 6 sheets in “main.xlsm” file. The Database sheet represents the dataset of accommodations. The Description sheet stores the definitions of columns of our dataset. The Temp sheet stores the values of the comboxes used in Visual Basic Program. The UserData sheet stores the input of a user. The default value of each row that stores data is “Irrelevant”, it is defined as -1; column D is used to convert values “TRUE” and “FALSE” to integers, i.e. 1 and 0. The Calculations sheet is used to calculate the score of each accommodation for the user. The Example sheet contains an example of how a continuous variable of the scoring function is calculated.

To add an example or variable, you will have to add them in both Database and Calculations sheets, in the last row if it an example and in the last column if it is a variable.

To change the calculation, go to the Calculations sheet:

1. Press the first cell of the column you want to change;
2. Change the calculation in the Formula Bar;
3. Double click the right angle of the cell with the updated formula.



*Figure 6. Excel formula in the cell of the first example of column Rooms.*

To change the scoring function go to column “BN” and repeat the aforementioned actions.

* 1. Changing Visual Basic Program

HomeSelector application contains 5 UserForms for *the Start page, the Main page, the Summary page, the Result page,* and *the Description page.* To create a user-friendly interface several types of controls were used: multipage, scrollbars (e.g. setting a price range), checkboxes (e.g. selecting “Irrelevant” option), option buttons (e.g. choosing a size of kitchen space, bathroom and living spaces), textboxes (e.g. specifying of an interval of square footage of total space), comboboxes (e.g. choosing a number of floors in a building), listbox (e.g. demonstrating results), images (e.g. adding a background for each userform).

To change the program you will need to:

1. Open the “main.xlsm” file;
2. Go to the “Developer” tab in Excel;
3. Press the “Visual Basic” button;
4. On the right on project explorer open the ”Forms” folder. You will see 5 user forms, they are used sequentially at different stages of the program.

The first user form represents *the Start Page* with the title of the program. The second is *the Main Page* where a user sets the options. The third is *the Summary page*: a user sees the options he or she chose and sets importance levels to the groups of answers. The fourth is used for displaying him the optimal examples (*the Result Page*). After choosing one of the examples a user sees the final fifth form (*The Description Page*) that shows the detailed information about the chosen accommodation.

5. To edit a Graphical user interface (GUI) of the userform double click it in the Forms folder. To view and edit the code of userforms, left-click the form and select *View Code.*

Let’s look at how the first variable *Price* is stored from GUI to the UserData sheet. To get to the code you can go to the second userform, select the first page if it is not selected and double click to one of two scroll bars of the range price. You will be directed to this piece of code:

1. **Private** **Sub** sb\_minprice\_Scroll()
2. **If** sb\_minprice.Value > sb\_maxprice.Value **Then**
3. sb\_maxprice.Value = sb\_minprice.Value
4. **End** **If**
5. updatePriceRange
6. **End** **Sub**
8. **Private** **Sub** sb\_maxprice\_Scroll()
9. **If** sb\_minprice.Value > sb\_maxprice.Value **Then**
10. sb\_minprice.Value = sb\_maxprice.Value
11. **End** **If**
12. updatePriceRange
13. **End** **Sub**

Two scroll bars set the range: sb\_minprice and sb\_maxprice. When these bars are scrolled functions sb\_minprice\_Scroll() and sb\_maxprice\_Scroll() are called. One can see that the if statements inside these functions won’t allow scroll bars to contradict, i.e. the value of sb\_maxprice can’t be smaller than the value of sb\_minprice and the other way around. After the if statement the function updatePriceRange() is executed.

1. **Private** **Sub** updatePriceRange()
2. l\_pricerange.Caption = Me.sb\_minprice.Value & " - " & Me.sb\_maxprice.Value & " $"
3. Sheets("UserData").Range("C2").Value = Me.sb\_minprice.Value
4. Sheets("UserData").Range("C3").Value = Me.sb\_maxprice.Value
5. **End** **Sub**

This function is responsible for setting *l\_pricerange* label in GUI and updating database values.

The same principle is applied to other elements: once the user interacts with the GUI element, the corresponding function will be called, which then will call another function that will update the database.

1. **Limitations**

There are a number of limitations of HomeSelector decision support system. Because of the different city structures, our Decision Support System cannot be applied in any city besides Moscow. For example, the Moscow metro is one of the largest in the world, so many apartments are relatively close to it, while in other cities of the world the metro can be far from apartments or it even may not exist. Further, HomeSelector does not take into account the time spent to get to the metro station or supermarket by car or by bicycle. Besides, there is no information about some of the most important customer requirements, namely, air pollution and security level that can strongly influence the choice of the buyer. Although the user can use the map in our system to determine the region of the apartment, however, it is not possible to set the exact address, a more detailed map (e.g. to build a route from work to an apartment) and find out information about the level of traffic in the region (Moscow has a high level of traffic jams, so the level of traffic is an important criterion). The next important limitation is the lack of functionality designed to display photos or layout of apartments, allowing the buyer to view the apartment in more detail. Last but not least, there is a lack of information about the infrastructure, for example, about schools, hospitals, parking that can be really useful to compare different offers and find the perfect accommodation.

1. **Conclusion**

To sum up, HomeSelector is a smart system supporting user-friendly interface and specially developed algorithm tofind the perfect accommodation based on the user`s preferences. Our Decision Support System has such components as identifying customer preferences, demonstration of selected characteristics, the output of results and general information about the apartments to make it more convenient for the buyer to find an apartment. The problems listed at the very beginning of this work were solved with the help of the developed system. Firstly, users can find perfect accommodation very fast by answering special questions, moreover, there is no longer a need to process a large amount of information, because the system is fully computerized and users can focus on details. Next, thanks to specially formulated questions, the user does not need to look at all existing apartments, since he or she can set the needed characteristics. And finally, an algorithm that we developed for our Decision Support System evaluates different apartments and displays suitable results based on user-defined preferences and relevance levels. Although the application has the limitations listed in the previous chapter, the developed functionality is basic Decision Support System of the real estate market that has general characteristics necessary for the choice of an apartment. This application can be used as an embedded component in already existing information systems in the real estate market of Moscow. In addition, further work is possible with a detailed examination of limitation and the development of special logical groups of questions for features of real estate markets in other cities.

1. **References**
2. Turban, E. and J. E. Aronson (2007). Decision Support Systems and Intelligent Systems (8th ed.). Prentice Hall Business Publishing.