

# White Box Testing

For each of our white box sets we decided to analyze one algorithm that we created for our KeaMatch Page.

## Set 1

### Default Algorithm

#### Source Code:

Both Algorithms expect User Model as a parameter, this model comes from Front-end and represents data from filled out form.

```
public Dictionary<UserModel, int> DefaultAlgorithm(UserModel MatchSeeker)
{
    //Potential Matches will be used in second Part of algorithm
    //result list servers as return object for frontend-storing the user and
percentage of "How good of a match they are for one another"- we wil call this number Match
Index

    //User Dummies are all curently registered users in our app
    List<UserModel> PotentialMatches = new List<UserModel>();
    Dictionary<UserModel, int> result = new Dictionary<UserModel, int>();
    List<UserModel> AllUsers = new UserDummies().GetUserDummies();

    int MatchSeekerAge = MatchSeeker.Age;

    foreach (var RegisteredUser in AllUsers)
    {
        var UserYear = RegisteredUser.Age;

        //if they are same age
        if (MatchSeekerAge == UserYear)
        {
            //add user with 50% Match Index
            result.Add(RegisteredUser, 50);

            //add user to list for futher evaluation
            PotentialMatches.Add(RegisteredUser);
        }
    }
}
```

```

    }
    else
    {
        //add user with 0% Match Index
        result.Add(RegisteredUser, 0);
    }

}
//if they match on age- check if they match on hobbies
foreach (var NarrowedDownUser in PotentialMatches)
{
    if (MatchSeeker.Hobby!=null)
    {
        var MatchSeekerHobbies = MatchSeeker.Hobby.Split(",");
        var Userhobbies = NarrowedDownUser.Hobby.Split(',');
        bool haveSharedHobbies = MatchSeekerHobbies.Intersect(Userhobbies).Count() > 0;
        if (haveSharedHobbies)
        {
            //the user is allready in result so we just change his Match Index
            result[NarrowedDownUser]=100;
        }
    }
}
return result;
}

```

## Pseudo Code:

Default Matching Algorithm;

potential matches = new(empty) list of users

result = new (empty) dictionary of users and their match percentage

AllUsers = registered users of our application

MatchSeekerYear = year of birth of the Matchseeker;

before\_check:

if(While not last user in All Users list){goto Second\_loop;}

get The year of user;

if(if Matchseeker Year equals

year of registered user)

{

```

add user to result with 50% match chance;
add user to potential matches;
}
else{add User to the result with 0% match chance; }
loop before_check;

```

Second\_loop:  
second\_check:

```

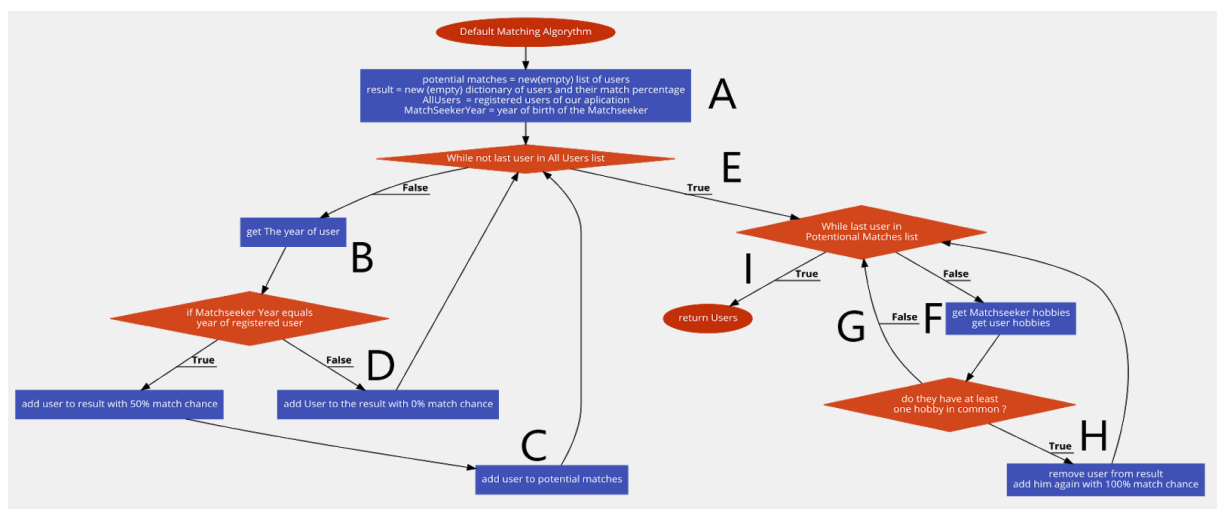
if(While last user in
Potential Matches list){ goto end }
get Matchseeker hobbies
get user hobbies;
if( do they have at least
one hobby in common ?)
{
remove user from result
add him again with 100% match chance;
}

```

goto second\_check

end:  
return return Users

Flow Chart:



## Minimal Paths for

100% Statement Coverage	100% Decision Coverage
A,B,D,E,F,G,I	A,B,C,B,D,E,F,H,F,G,I

## Set 2

## Advanced Algorithm

Source Code:

```
public Dictionary<UserModel, int> AdvancedAlgorithm(UserModel
MatchSeeker)
{
    //result list servers as return object for frontend-storing
the user and percentage of
    //"How good of a match they are for one another"- we wil
call this number Match Index
    //Lists for matches on different values are created
    //User Dummies are all curently registered user in our app
    Dictionary<UserModel, int> result = new
Dictionary<UserModel, int>();
    List<UserModel> SameGenderMatches = new List<UserModel>();
    List<UserModel> SameHeightMatches = new List<UserModel>();
    List<UserModel> SameSignMatches = new List<UserModel>();

    List<UserModel> AllUsers = new
UserDummies().GetUserDummies();

    foreach (var RegisteredUser in AllUsers)
    {

        if (RegisteredUser.Gender==MatchSeeker.Gender)
        {
            SameGenderMatches.Add(RegisteredUser);
        }
        if(RegisteredUser.Height==MatchSeeker.Height)
```

```

        {
            SameHeightMatches.Add(RegisteredUser);
        }
        if (RegisteredUser.Zodiac == MatchSeeker.Zodiac)
        {
            SameSignMatches.Add(RegisteredUser);
        }
        else
        {
            //If both the matchseeker and user don't match
            //add user to the result with 0 match index
            result.Add(RegisteredUser, 0);
        }
    }
    //We wil use gender interest as a main factor in this
    algorithm
    foreach (var match in SameGenderMatches)
    {
        bool isInHeightMatches =
        SameHeightMatches.Contains(match);
        bool isInzodiacMatches =
        SameSignMatches.Contains(match);
        //lets check if they match on all values
        //otherwise check if they match at least on one value
        if (isInHeightMatches && isInzodiacMatches)
        {
            result.Add(match, 100);
        }
        else if (isInHeightMatches || isInzodiacMatches)
        {
            //if the user is allready in the list change his
            Match Index
            if (result.ContainsKey(match))
            {
                result[match] = 50;
            }
            else
            {
                result.Add(match, 50);
            }
        }
    }
}

```

```
        return result;
    }
```

## Pseudo Code:

Advanced Matching Algorithm;

result = new (empty) dictionary of users and their match percentage

same gender matches = new(empty) list of users

same height matches = new(empty) list of users

same sigh matches = new(empty) list of users

AllUsers = registered users of our aplication

{

first\_check:

if(While not last user in AllUsers list)

{

Second\_loop:

second\_check:

if(While not last user in the Gender list){ goto end1 }

else if(Do their heights AND zodiacs match?)

Add them to the result list with 100% match chance;

else if(Do their heights OR zodiacs match?)

Add them to the result list with 50% match chance;

else

Add them to the result list with 0% match chance;

goto second\_check

end1:

return Return result Users;

}

else if(Do their genders match?)

Add them to the same gender list;

else if(Do their heights match?)

Add them to the same height list;

else if(Do their zodiacs match?)

Add them to the same zodiac list;

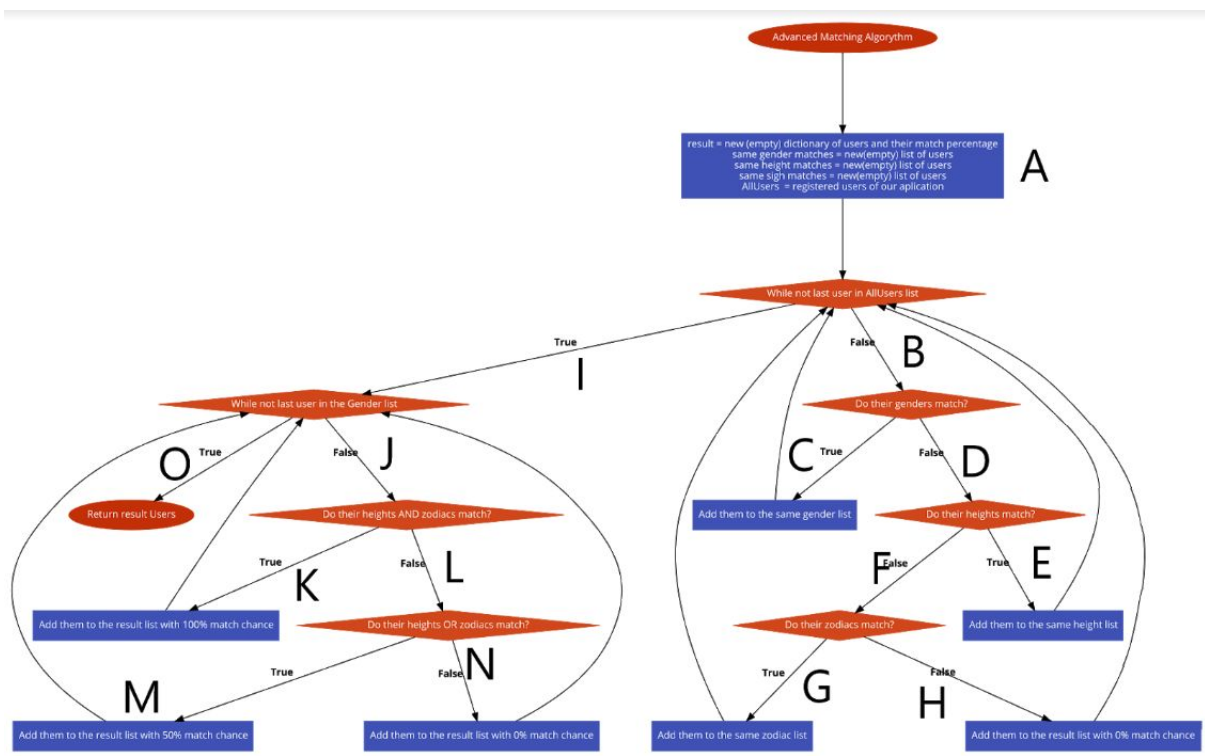
else

Add them to the result list with 0% match chance;

goto first\_check

}

# Flow Chart:



## Minimal Paths for

100% Statement Coverage	100% Decision Coverage
A,B,D,F,G,I,J,L,N	A,B,C,B,D,E,B,D,F,G,B,D,F,H,I,J,K,J,L, M,J,L,N,O