```
Script started on Wed 15 Mar 2017 02:20:16 PM CDT
\033]0;b_pepa@mars:~/CSC122/copy_xyz\007[b_pepa@mars copy_xyz]$ pwd
/home/students/b_pepa/CSC122/copy_xyz
\033]0;b_pepa@mars:~/CSC122/copy_xyz\007[b_pepa@mars copy_xyz]$ cat copy.info
Brandon Pepa
CSC122-001
Copy XYZ
Lab

(level 1)
(level 1.5)
        fix user data being longer than declared cstring length
(level 1)
        allow blank lines between data groups
(level 1.5)
        allow the user to place whole line comments in data file
(level 1)
        allow end of line comments as well

**(level 6)**

Description:
        The function of this program is to copy the data from one file which has data
on
students name, id, gpa, and grade to another file. the data has to be ordered correctl
y
in order for the data to transfer properly.
\033]0;b_pepa@mars:~/CSC122/copy_xyz\007[b_pepa@mars copy_xyz]$ cat copy.cpp
#include <iostream>
#include <fstream>
#include <cctype>
#include <cstring>
using namespace std;

const long MAX_NAME = 200;
const long MAX_FNAME = 200;

class student
{
   char name[MAX_NAME];
   long id;
   double gpa;
   char grade;
public:
   //constructors
   student(void) : name(), id(0), gpa(0.0), grade('F') {}

   student(const student & other) : id(other.id), gpa(other.gpa),
           grade(other.grade) { strcpy(name, other.name); }

   student(const char new_name[MAX_NAME], const long new_id,
           const double new_gpa, const char new_grade) : id(new_id),
           gpa(new_gpa), grade(new_grade) { strcpy(name, new_name); }

   //accesors
   void get_name(char other[MAX_NAME]) const { strcpy(other, name); return; }
   long get_id(void) const { return id; }
   double get_gpa(void) const { return gpa; }
   char get_grade(void) const { return grade; }

   //mutators
   void set_name(const char other[MAX_NAME]) { strcpy(name, other); return; }
   void set_id(const long new_id) { id = new_id; return; }
   void set_gpa(const double new_gpa) { gpa = new_gpa; return; }
   void set_grade(const char new_grade) { grade = new_grade; return; }

   void read(ifstream & f);
   void write(ofstream & f) const;

};

//This function removes any spaces and comments before the
//next piece of data in a stream
void rm_SC(ifstream & f);



int main(void)
{
   //input and output streams linked to a file
   ifstream infile;
   ofstream outfile;

   //cstring variable for storing the name of the input and output file
   char fname[MAX_FNAME];
   char fname2[MAX_FNAME];

   //stores the data of the student before writing
   //to the output file
   student student;

   cout << "\tWelcome to the People Data Copying Program!";

   //Opens the input file
   cout << "\n\nPlease enter the name of your data file: ";
   cin.getline(fname, MAX_FNAME);
   infile.open(fname);
   while(!infile)
   {
      infile.close();
      infile.clear();
      cout << "\nI'm sorry, I could not open \'" << fname << "\'."
              " Please enter another name:\n";
      cin.getline(fname, MAX_FNAME);
      infile.open(fname);
   }

   cout << "File \'" << fname << "\' opened successfully!";

   //Opens the file to copy to
   cout << "\n\nPlease enter the name of the copy file: ";
   cin.getline(fname2, MAX_FNAME);
   outfile.open(fname2);
   while(!outfile)
   {
      outfile.close();
      outfile.clear();
      cout << "\nI'm sorry, I could not open \'" << fname2 << "\'. "
              " Please enter another name:\n";
      cin.getline(fname2, MAX_FNAME);
      outfile.open(fname2);
   }

   cout << "File \'" << fname2 << "\' opened successfully!"
           "\n\nCopying data from \'" << fname << "\' to \'" << fname2
        << "\'...";

   //reads the file until the eof character
   infile.peek();
   while(!infile.eof())
   {
      student.read(infile);
      student.write(outfile);
      infile.peek();
   }

   cout << "\n\nDone copying data!"
           "\n\nThank you for using the data copy program!!\n\n";

return 0;
}

void student::read(ifstream & f)
{
   rm_SC(f);

   f.getline(name, MAX_NAME);

   //removes any comments from the end of the cstring
   size_t i = 0;
   while(name[i] != '\0')
   {
      ++i;

      //if there is a comment, set that character to the null character
      if(name[i] == '#')
      {
        name[i] = '\0';
      }
   }

   rm_SC(f);

   //if the name is too long, ignore the rest
   if(!isdigit(f.peek()))
   {
      f.ignore(INT_MAX, '\n');
   }

   //After each input of data, any comments or spacing is removed
   //so the next piece of data garuenteed to read
   f >> id;

   rm_SC(f);

   f >> gpa;

   rm_SC(f);

   f >> grade;

   rm_SC(f);

return;
}

void student::write(ofstream & f) const
{
   f << name << endl << id << endl << gpa
     << endl << grade << endl;
return;
}
```

```

void rm_SC(ifstream & f)
{
   while(isspace(f.peek()) || f.peek() == '#')
   {
      if(isspace(f.peek()))
      {
         f.ignore();
      }
      else
      {
         f.ignore(INT_MAX, '\n');
      }
   }
}
\033]0;b_pepa@mars:~/CSC122/copy_xyz\007[b_pepa@mars copy_xyz]$ CPP copy
copy.cpp***


\033]0;b_pepa@mars:~/CSC122/copy_xyz\007[b_pepa@mars copy_xyz]$ vi\033[K033[K033[Kcat
students
Jason James
123456
8.2
B




Tammy James #end of line comment
123462 #comment
3.6 #comments
A #comments and stuff
#comment on it's own line
Brandon Pepa
52341
4.8
C
John Doe
67192     3.2  R
#unneccessary spaces
Ryan
01392  #end of line comment
1.2
G
Michael Scott
98234
5.2
T #idk what else I can do to test this
\033]0;b_pepa@mars:~/CSC122/copy_xyz\007[b_pepa@mars copy_xyz]$ ./copy.out
        Welcome to the People Data Copying Program!

Please enter the name of your data file: bob.dat

I'm sorry, I could not open 'bob.dat'. Please enter another name:
students
File 'students' opened successfully!

Please enter the name of the copy file: /cant read this

I'm sorry, I could not open '/cant read this'.  Please enter another name:
students.bak
File 'students.bak' opened successfully!

Copying data from 'students' to 'students.bak'...
```

```

Done copying data!

Thank you for using the data copy program!!

\033]0;b_pepa@mars:~/CSC122/copy_xyz\007[b_pepa@mars copy_xyz]$ cat students.bak
Jason James
123456
8.2
B
Tammy James
123462
3.6
A
Brandon Pepa
52341
4.8
C
John Doe
67192
3.2
R
Ryan
1392
1.2
G
Michael Scott
98234
5.2
T
\033]0;b_pepa@mars:~/CSC122/copy_xyz\007[b_pepa@mars copy_xyz]$ va\033[K\033[K\033[Kcat
copy.tpq
Thought Provoking Questions

1. What weird behavior does open exhibit for output files by default? How
   do we fix this problem?

A) output files will be truncated by default so all the data already in the
   file will be lost if we write our own data into it. We fix this problem
   by protecting the output file. if there's data in it we give the user
   the choice to overwrite, append or choose another file name

2. How much does spacing matter in the input file? The output file? (Hint:
   would it matter if it weren't present at all? If there were many, many
   space?)

A) spacing is important to a certain extent. normal input will ignore
   spaces between data (just like cin) unless you use getline(). obviously
   there needs to be spaceing between data of the same type (except
   characters) in order to distinguish between data.

3. Problems with the (C)string piece of data:
  i. What problem might you have with the (C)string data (being as it is
     'mixed' with so many other data types in this file: number and
     characters and such)? (Hint: Is the (C)string data one or multiple
     words?) Is this difficult to fix? What assumptions did you make to solv
     this paroblem?

A) the cstring data is usually multiple words (it's a name of a person). we
   can assume it's always on one line and there's no other data in that line
   so we'll use the the getline() function to input the cstring data.

  ii. If the (C)string had to be placed after the other data -- at the end
      of the data group/block, what problem might arise? How do we
      typically avoid this situation (again, assuming the data has to be
      in that order)? [assume you have rewritten your code to deal with the
      new data order -- but do not do so.]

A) I don't know... :/

  iii. Think about, but to not fix, the potential problem of the user's
       (C)string being longer than you had anticipated. (Answer this question
       even if you used the string class to code your program!)

A) When using the getline function to get the name of the person, the second
   argument will be the max length of your cstring. The rest of the data can
   be thrown out if the name is too long?

4i.  What function is used to tell when you've reached the end of a stream?

A) The "End of File" (eof) character is at the end of every file so it can be
   used in a sentinal loop. (file_name.eof())

  ii. Can this function be used on the keyboard stream?

A) This function CANNOT be used on keyboard stream becuse the eof charater
   causes an un-clearable error for cin.

5. How do you pass a stream to a function?

6. Why is it a good idea to make input functions ignorant of wheather or
   not a stream is cin or a file? Output functions/cout/file?

7. Why do we close files?

A) open files use system resorces and an open file adds to the limit on the
   operating system for max number of files open. The output buffer is
   flushed when you close a file also so anything remaining in your buffer
   will be put into the file.
\033]0;b_pepa@mars:~/CSC122/copy_xyz\007[b_pepa@mars copy_xyz]$ exit
exit

Script done on Wed 15 Mar 2017 02:24:39 PM CDT
```