String extended

Just a partial solution:

```
#include<iostream>

using namespace std;

class String {
private:
  char *str;
  int length;
  int c_string_length(const char* s);
public:
  String();
  String(const char* s);
  String(const String& other);
  String& operator=(String& other);

  void print();
  int getLength();

  String operator+(String other);

  friend String operator+(const char* left, String right);
  friend ostream& operator<<(ostream& os, String s);

  ~String();
};

int String::c_string_length(const char* s) {
  int i;
  for(i = 0; s[i] != '\0'; i++) {}
  return i;
}

String::String() {
  str = new char[1];
  str[0] = '\0';
  length = 0;
}

String::String(const char* s) {
  length = c_string_length(s);
  str = new char[length + 1];
  for(int i = 0; s[i] != '\0'; i++) {
    str[i] = s[i];
  }
  str[length] = '\0';
}

String::String(const String& other) {
  this->length = other.length;
  str = new char[this->length + 1];
  for(int i = 0; other.str[i] != '\0'; i++) {
    this->str[i] = other.str[i];
  }
  this->str[this->length] = '\0';
}

String& String::operator=(String& other) {
  if(this == &other) {
    return *this;
  }
  delete[] this->str;
  this->str = new char[other.length];
  this->length = other.length;
  for(int i = 0; other.str[i] != '\0'; i++) {
    this->str[i] = other.str[i];
  }
  this->str[this->length] = '\0';
  return *this;
}

void String::print() {
  cout << str << endl;
}

int String::getLength() {
  return length;
}

String String::operator+(String other) {
  int new_length = this->length + other.length + 1;
  char s[new_length];
  for(int i = 0; i < this->length; i++) {
    s[i] = this->str[i];
  }
  for(int i = 0; i < other.length; i++) {
    s[i + this->length] = other.str[i];
  }
  s[new_length] = '\0';
  return String(s);
}

String::~String() {
  delete[] str;
}

String operator+(const char* left, String right) {
  int j;
  for(j = 0; left[j] != '\0'; j++) {}
  int new_length = j + right.length + 1;
  char s[new_length];
  for(int i = 0; i < j; i++) {
    s[i] = left[i];
  }
  for(int i = 0; i < right.length; i++) {
    s[i + j] = right.str[i];
  }
  s[new_length] = '\0';
  return String(s);
}

ostream& operator<<(ostream& os, String s) {
  os << s.str;
  return os;
}


int main(void) {
  String s1 = String("batman");
  String s2 = String("catman");
  cout << s1 << endl;
  cout << s2 << endl;
  //cout << (s1 == s2) << endl;
  s1 = s2;
  //cout << (s1 == s2) << endl;
  cout << s1 << endl;
  cout << s2 << endl;
  return 0;
}
```

Grades (not done)

```
#include<iostream>

using namespace std;

class String {
private:
  char *str;
  int length;
  int c_string_length(const char* s);
public:
  String();
  String(const char* s);
  String(const String& other);
  String& operator=(String& other);
  //String& operator=(const char* s);

  void print();
  int getLength();

  String operator+(String other);
  bool operator==(String& other);
  //bool operator==(const char* s);

  friend String operator+(const char* left, String right);
  friend ostream& operator<<(ostream& os, String s);

  ~String();
};

int String::c_string_length(const char* s) {
  int i;
  for(i = 0; s[i] != '\0'; i++) {}
  return i;
}

String::String() {
  str = new char[1];
  str[0] = '\0';
  length = 0;
}

String::String(const char* s) {
  length = c_string_length(s);
  str = new char[length + 1];
  for(int i = 0; s[i] != '\0'; i++) {
    str[i] = s[i];
  }
  str[length] = '\0';
}

String::String(const String& other) {
  this->length = other.length;
  str = new char[this->length + 1];
  for(int i = 0; other.str[i] != '\0'; i++) {
    this->str[i] = other.str[i];
  }
  this->str[this->length] = '\0';
}

String& String::operator=(String& other) {
  if(this == &other) {
    return *this;
  }
  delete[] this->str;
  this->str = new char[other.length];
  this->length = other.length;
  for(int i = 0; other.str[i] != '\0'; i++) {
    this->str[i] = other.str[i];
  }
  this->str[this->length] = '\0';
  return *this;
}

void String::print() {
  cout << str << endl;
}

int String::getLength() {
  return length;
}

String String::operator+(String other) {
  int new_length = this->length + other.length + 1;
  char s[new_length];
  for(int i = 0; i < this->length; i++) {
    s[i] = this->str[i];
  }
  for(int i = 0; i < other.length; i++) {
    s[i + this->length] = other.str[i];
  }
  s[new_length] = '\0';
  return String(s);
}

bool String::operator==(String& other) {
  if (this->length != other.length) {
    return false;
  }
  for(int i = 0; i < this->length; i++) {
    if(this->str[i] != other.str[i]) {
      return false;
    }
  }
  return true;
}

String::~String() {
  delete[] str;
}

String operator+(const char* left, String right) {
  int j;
  for(j = 0; left[j] != '\0'; j++) {}
  int new_length = j + right.length + 1;
  char s[new_length];
  for(int i = 0; i < j; i++) {
    s[i] = left[i];
  }
  for(int i = 0; i < right.length; i++) {
    s[i + j] = right.str[i];
  }
  s[new_length] = '\0';
  return String(s);
}

ostream& operator<<(ostream& os, String s) {
  os << s.str;
  return os;
}


class Grades {
private:
  String names[100];
  int points[100];
  int length;
public:
  Grades();
  void add(const char* n, int p);
};

Grades::Grades() {
  length = 0;
}


void Grades::add(const char* n, int p) {
  String s = String(n);
  names[length] = s;
  points[length] = p;
  length++;
}

int main(void) {
  Grades grades;
  grades.add("Andras", 56);
  grades.add("Aladar", 22);
  grades.add("Anita", 71);
  grades.add("Andrea", 34);
  grades.add("Aniko", 64);
  /*
  for(int i = 0; i < grades.length(); i++) {
    cout << grades[i] << " : " << grades[grades[i]] << endl;
  }
  cout << "avg: " << grades.avg() << endl;
  */
  return 0;
}
```