HOMEWORK 4

Grade id: 17

1.-Implement the class clock type described in the slides, use the same const modifier for all the methods FOR THE METHODS THAT HAVE THE CONST MODIFIER, the ones that don't DO NOT ADD it.

```
class clockType
{
public:
   clockType(){
      hr = 00;
      mini = 00;
      sec = 00;
   }
   void setTime(int hours, int minutes, int seconds){
      if(hours <=23 && hours >=0) hr = hours;
      else hr =00;
      if(minutes <=59 && minutes >=0) mini = minutes;
      else mini =00;
      if(seconds <=59 && seconds >=0) sec = seconds;
      else sec =00;
   }
   void getTime(int& hours, int& minutes, int& seconds) const{
      hours = hr;
      minutes = mini;
      seconds = sec;
   }
   void printTime() const{
      cout << hr << ":" << mini << ":" << sec << endl;
   }
   void incrementSeconds(){
      if(sec<59) sec++;
      else{
         sec=00;
         incrementMinutes();
      }
   }
   void incrementMinutes(){
      if(mini<59) mini++;
         else{
            mini=00;
            incrementHours();
         }
   }
   void incrementHours(){
      if(hr<23) hr++;
         else{
            hr=00;
         }
   }
   bool equalTime(const clockType& otherClock) const{
      if(hr == otherClock.hr && mini == otherClock.mini && sec == otherClock.sec)
```

```
        return 1;
    else
        return 0;
  }
private:
  int hr;
  int mini;
  int sec;
};
```

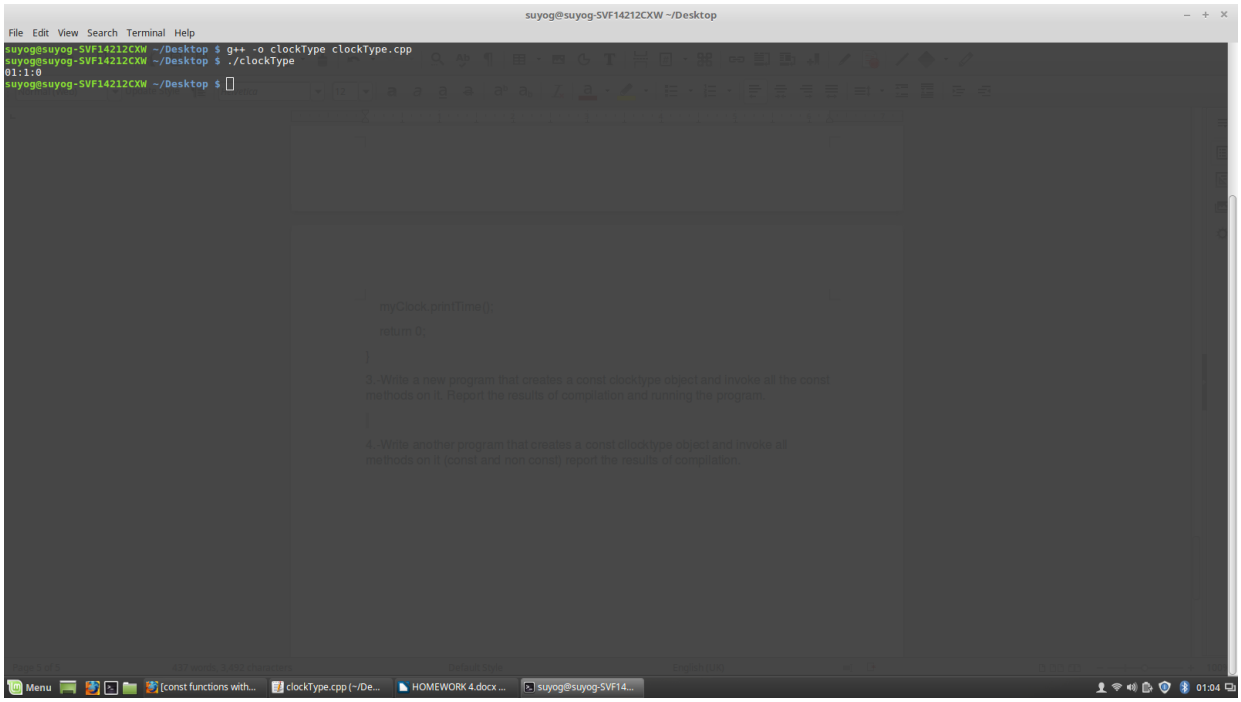
2.-Write a program that creates a clocktype object and calls all the methods. Report the results of compiling and running the program.

```
#include <iostream>
using namespace std;

class clockType
{
public:
  clockType(){
    hr = 00;
    mini = 00;
    sec = 00;
  }
  void setTime(int hours, int minutes, int seconds){
    if(hours <=23 && hours >=0) hr = hours;
    else hr =00;
    if(minutes <=59 && minutes >=0) mini = minutes;
    else mini =00;
    if(seconds <=59 && seconds >=0) sec = seconds;
    else sec =00;
  }
  void getTime(int& hours, int& minutes, int& seconds) const{
    hours = hr;
    minutes = mini;
    seconds = sec;
  }
  void printTime() const{
    cout << hr << ":" << mini << ":" << sec << endl;
  }
  void incrementSeconds(){
    if(sec<59) sec++;
    else{
      sec=00;
      incrementMinutes();
    }
  }
  void incrementMinutes(){
    if(mini<59) mini++;
      else{
        mini=00;
        incrementHours();
```

```
        }
    }
    void incrementHours(){
        if(hr<23) hr++;
            else{
                hr=00;
            }
    }
    bool equalTime(const clockType& otherClock) const{
        if(hr == otherClock.hr && mini == otherClock.mini && sec == otherClock.sec)
            return 1;
        else
            return 0;
    }
private:
    int hr;
    int mini;
    int sec;
};

int main()
{
    int hours=00;
    int minutes=00;
    int seconds=00;
    clockType myClock;
    const clockType otherClock;
    myClock.setTime(23,59,59);
    myClock.getTime(hours,minutes,seconds);
    myClock.incrementHours();
    myClock.incrementMinutes();
    myClock.incrementSeconds();
    myClock.equalTime(otherClock);
    myClock.printTime();
    return 0;
}
```



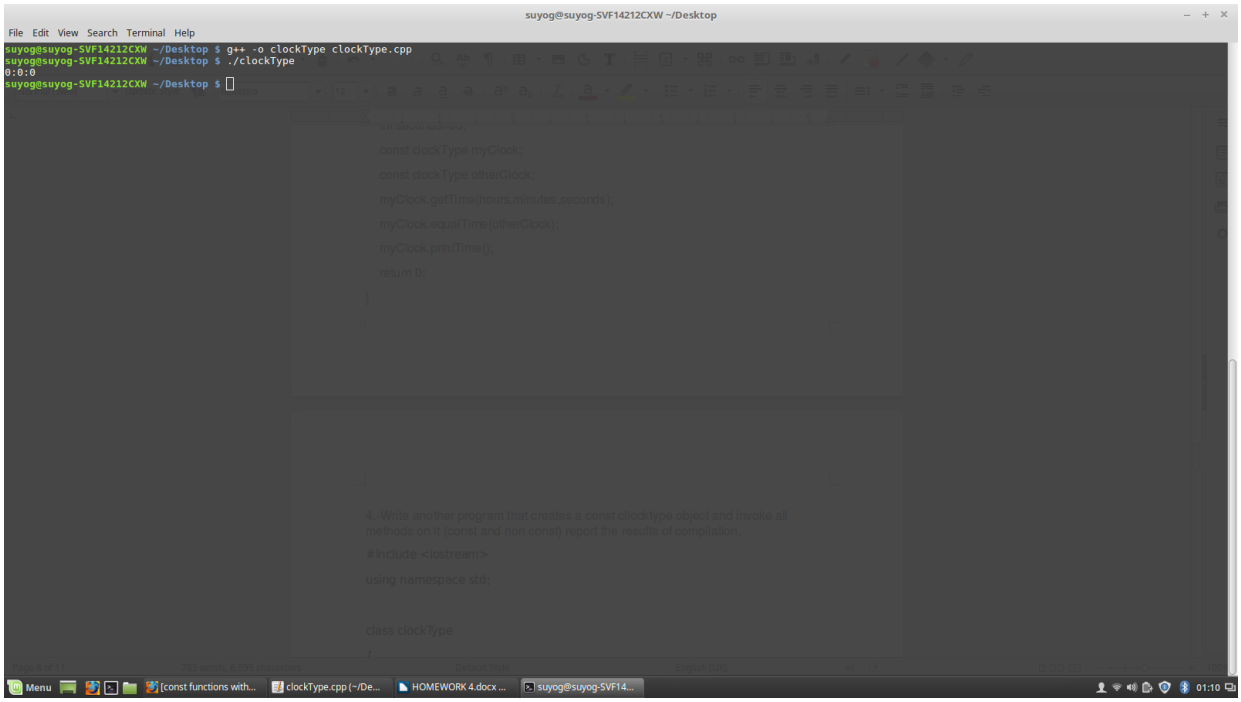
3.-Write a new program that creates a const clocktype object and invoke all the const methods on it. Report the results of compilation and running the program.

```
 #include <iostream>
using namespace std;

class clockType
{
public:
   clockType(){
      hr = 00;
      mini = 00;
      sec = 00;
   }
   void setTime(int hours, int minutes, int seconds){
      if(hours <=23 && hours >=0) hr = hours;
      else hr =00;
      if(minutes <=59 && minutes >=0) mini = minutes;
      else mini =00;
      if(seconds <=59 && seconds >=0) sec = seconds;
      else sec =00;
   }
   void getTime(int& hours, int& minutes, int& seconds) const{
      hours = hr;
      minutes = mini;
      seconds = sec;
   }
   void printTime() const{
      cout << hr << ":" << mini << ":" << sec << endl;
   }
   void incrementSeconds(){
      if(sec<59) sec++;
```

```
      else{
         sec=00;
         incrementMinutes();
      }
   }
   void incrementMinutes(){
      if(mini<59) mini++;
         else{
            mini=00;
            incrementHours();
         }
   }
   void incrementHours(){
      if(hr<23) hr++;
         else{
            hr=00;
         }
   }
   bool equalTime(const clockType& otherClock) const{
      if(hr == otherClock.hr && mini == otherClock.mini && sec == otherClock.sec)
         return 1;
      else
         return 0;
   }
private:
   int hr;
   int mini;
   int sec;
};

int main()
{
   int hours=00;
   int minutes=00;
   int seconds=00;
   const clockType myClock;
   const clockType otherClock;
   myClock.getTime(hours,minutes,seconds);
   myClock.equalTime(otherClock);
   myClock.printTime();
   return 0;
}
```



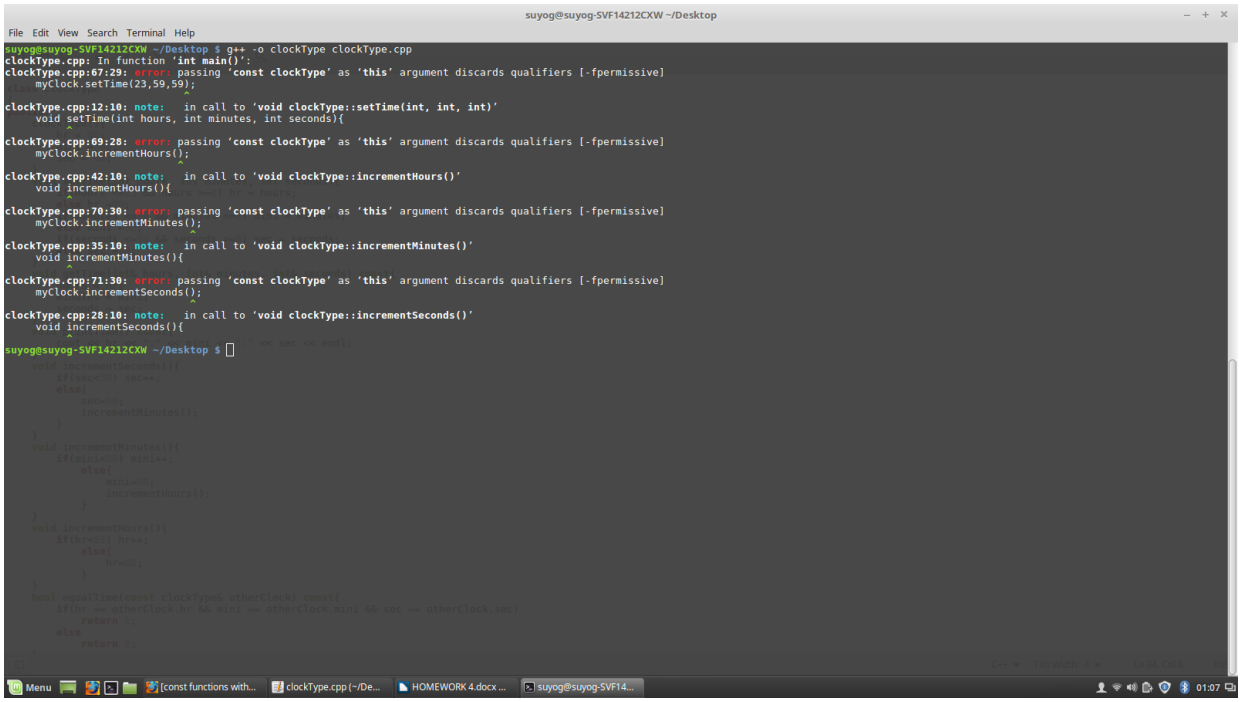
4.-Write another program that creates a const cllocktype object and invoke all methods on it (const and non const) report the results of compilation.

```
#include <iostream>
using namespace std;

class clockType
{
public:
   clockType(){
      hr = 00;
      mini = 00;
      sec = 00;
   }
   void setTime(int hours, int minutes, int seconds){
      if(hours <=23 && hours >=0) hr = hours;
      else hr =00;
      if(minutes <=59 && minutes >=0) mini = minutes;
      else mini =00;
      if(seconds <=59 && seconds >=0) sec = seconds;
      else sec =00;
   }
   void getTime(int& hours, int& minutes, int& seconds) const{
      hours = hr;
      minutes = mini;
      seconds = sec;
   }
   void printTime() const{
      cout << hr << ":" << mini << ":" << sec << endl;
   }
   void incrementSeconds(){
      if(sec<59) sec++;
      else{
```

```
        sec=00;
        incrementMinutes();
      }
   }
   void incrementMinutes(){
      if(mini<59) mini++;
        else{
          mini=00;
          incrementHours();
        }
   }
   void incrementHours(){
      if(hr<23) hr++;
        else{
          hr=00;
        }
   }
   bool equalTime(const clockType& otherClock) const{
      if(hr == otherClock.hr && mini == otherClock.mini && sec == otherClock.sec)
        return 1;
      else
        return 0;
   }
private:
   int hr;
   int mini;
   int sec;
};

int main()
{
   int hours=00;
   int minutes=00;
   int seconds=00;
   const clockType myClock;
   const clockType otherClock;
   myClock.setTime(23,59,59);
   myClock.getTime(hours,minutes,seconds);
   myClock.incrementHours();
   myClock.incrementMinutes();
   myClock.incrementSeconds();
   myClock.equalTime(otherClock);
   myClock.printTime();
   return 0;
}
```


suyog@suyog-SVF14212CXW ~/Desktop
File Edit View Search Terminal Help
suyog@suyog-SVF14212CXW ~/Desktop $ g++ -o clockType clockType.cpp
clockType.cpp: In function 'int main()':
clockType.cpp:67:29: error: passing 'const clockType' as 'this' argument discards qualifiers [-fpermissive]
myClock.setTime(23,59,59);
clockType.cpp:12:10: note: in call to 'void clockType::setTime(int, int, int)'
void setTime(int hours, int minutes, int seconds){
clockType.cpp:69:28: error: passing 'const clockType' as 'this' argument discards qualifiers [-fpermissive]
myClock.incrementHours();
clockType.cpp:42:10: note: in call to 'void clockType::incrementHours()'
void incrementHours(){
clockType.cpp:70:30: error: passing 'const clockType' as 'this' argument discards qualifiers [-fpermissive]
myClock.incrementMinutes();
clockType.cpp:35:10: note: in call to 'void clockType::incrementMinutes()'
void incrementMinutes(){
clockType.cpp:71:30: error: passing 'const clockType' as 'this' argument discards qualifiers [-fpermissive]
myClock.incrementSeconds();
clockType.cpp:28:10: note: in call to 'void clockType::incrementSeconds()'
void incrementSeconds(){
suyog@suyog-SVF14212CXW ~/Desktop $
Menu
[const functions with...
clockType.cpp (~/De...
HOMEWORK 4.docx ...
suyog@suyog-SVF14...
01:07