```
#! /usr/bin/python
from PyQt4.QtGui import QTextDocument, QPrinter, QApplication
import os
import sys
import logging
import argparse
import tempfile
import subprocess


def logger(func):
    def log_wrap(self, ifile=None, ofile=None, size="A4"):
        logging.getLogger().name = "py2pdf> "
        logging.getLogger().setLevel(logging.INFO)
        func(self, ifile, ofile, size)
    return log_wrap


class Py2pdf:

    """
        converts python file to pdf
    """
    @logger
    def __init__(self, ifile=None, ofile=None, size="A4"):
        self.size = size
        if not ifile:
            raise Exception("input file is required")
        else:
            self.py_file = ifile
            if ofile:
                self.pdf_file = ofile
            else:
                self.pdf_file = ifile.split('.')[0]+".pdf"
            with tempfile.NamedTemporaryFile(delete=False, suffix=".py") as temp:
                self.temp_file = temp.name
                temp_data = open(self.py_file, 'rb').read()
                temp.write(temp_data)
                temp.flush()

    def init_print(self):
        app = QApplication(sys.argv)
        doc = QTextDocument()
        temp_html = self.temp_file[:-2] + "html"
        r = "pygmentize -O full,style=emacs -f html -l python -o %s %s" % (
            temp_html, self.py_file)
        s = subprocess.Popen(r, shell=True)
        s.wait()
        html = open(temp_html).read()
        doc.setHtml(html)
        printer = QPrinter()
        printer.setOutputFileName(self.pdf_file)
        printer.setOutputFormat(QPrinter.PdfFormat)
        if self.size.lower() == "a2":
            printer.setPageSize(QPrinter.A2)
        elif self.size.lower() == "a3":
            printer.setPageSize(QPrinter.A3)
        elif self.size.lower() == "a4":
            printer.setPageSize(QPrinter.A4)

        printer.setPageMargins(15, 15, 15, 15, QPrinter.Millimeter)
        doc.print_(printer)
        logging.info("PDF created at %s" % (self.pdf_file))
```

```python
def parse_arg():
    parser = argparse.ArgumentParser(description=" \
             Convert Python code into pdf with syntax highlighting", epilog="\
             Author:tushar.rishav@gmail.com")
    parser.add_argument(
        "-i", "--ifile", help="Absolute path of the python file", type=str)
    parser.add_argument(
        "-o", "--ofile", help="Absolute path of the output pdf file", type=str)
    parser.add_argument(
        "-s", "--size", help="PDF size. A2,A3,A4,A5 etc", type=str, default="A3")
    args = parser.parse_args()
    if not args.ifile:
        raise Exception("input file is required")
    else:
        ifile = args.ifile
        if args.ofile:
            pdf_file = args.ofile
        else:
            pdf_file = args.ifile.split('.')[0]+".pdf"
    return (ifile, pdf_file, args.size)

def main():

    ifile, ofile, size = parse_arg()
    pdf = Py2pdf(ifile, ofile, size)
    pdf.init_print()

if __name__ == "__main__":
    main()
```