《程序设计基础》课程

实验指导书

东北大学软件学院

2017 年 5 月

目　录

实验一　选择结构程序设计

1.1　实验内容

1. 熟悉 RAPTOR 算法设计环境。设计一个程序，输出一个提示信息，请求用户输入一个数字，如果大于 0 则输出“Positive”，如果小于 0 则输出“Negative”，如果等于 0 则输出“Zero”。

2. 根据用户输入的应征税收入计算所得税，下表给出了相关数据，确保程序中包含错误检查部分以防用户输入负数。

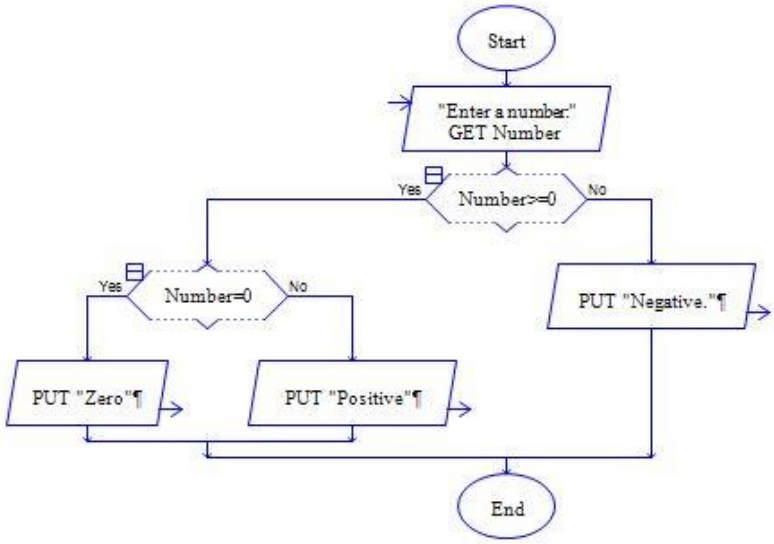
应征税收入		所得税
起始	截止	
0	50000	0 以上的 5%
50000	100000	2500+50000 以上的 7%
100000	……	6000+100000 以上的 9%

3. 输入某年某月某日，判断这一天是这一年的第几天？如：2000 年 3 月 1 日就是 2000 年的第 61 天 31+29+1 从 1 计数）

1.2　实验目的与要求

了解所使用的计算机系统和在该系统上如何进行程序设计；掌握选择结构的程序设计方法；编辑和运行 Python 程序。实验要求采用选择结构进行程序设计，给出程序设计流程图或伪代码，并使用 Python 代码编程实现，且编写实验报告。

1.3 RAPTOR 参考流程图



1.4 C++参考实现代码

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
stringraptor_prompt_variable_zzyz;
int number;

raptor_prompt_variable_zzyz ="Enter a number:";
cout<<raptor_prompt_variable_zzyz<<endl;
cin>> number;
if (number>=0)
   {if (number==0)
{cout<< "Zero" <<endl;           }
   else
{cout<< "Positive" <<endl;       }
     }
else
     {cout<< "Negative." <<endl;      }
return 0;
}
```

1.5 PYTHON 参考实现代码

```
# -*- coding:utf-8 -*-
number = int(input("Enter a number:"))
if (number>=0):
if (number==0):
print("Zero")
else:
print("Positive")
else:
print("Negative")
```

实验二　循环结构程序设计

2.1　实验内容

1. 设计一个程序，从键盘输入 5 个整数，计算累加和，将计算结果输出到屏幕上。修改程序设计，询问用户进行多少个整数的累加和，如果超过 10 个，则提示重新输入，否则计算累加和并且输出计算结果。

2. 从控制台输入一行字符，分别统计出其中英文字母、空格、数字和其它字符的个数。

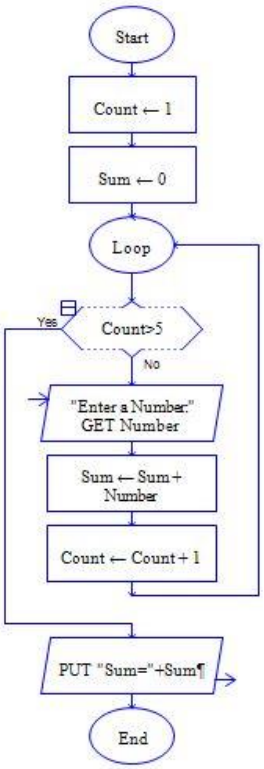
3. 古典问题：有一对兔子，从出生后第 3 个月起每个月都生一对兔子，小兔子长到第三个月后每个月又生一对兔子，假如兔子都不死，问：从第三个月到第二十个月，每个月的兔子总数为多少？

提示：兔子的规律为数列 1,1,2,3,5,8,13,21....

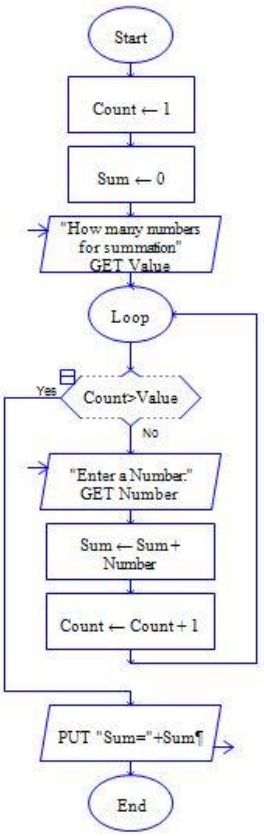
2.2　实验目的与实验要求

掌握循环结构的程序设计方法；编辑和运行 Python 程序。实验要求采用循环结构进行程序设计，给出程序设计流程图或伪代码，并使用 Python 代码编程实现，且编写实验报告。

2.3 RAPTOR 参考流程图



修改后：



2.4 C++参考实现代码

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
stringraptor_prompt_variable_zzyz;
int count;
int sum;
int number;
int value;
count =1;
sum =0;
raptor_prompt_variable_zzyz ="How many numbers for summation";
cout<<raptor_prompt_variable_zzyz<<endl;
cin>> value;
while (!(count>value))
  {raptor_prompt_variable_zzyz ="Enter a Number:";
cout<<raptor_prompt_variable_zzyz<<endl;
cin>> number;
sum =sum+number;
count =count+1;      }
cout<< "Sum="<< sum <<endl;
return 0;
}
```

2.5 PYTHON 参考实现代码

```
# -*- coding:utf-8 -*-

count=1

sum=0

while(1):

if count>5:

break

else:

number = int(input("Enter a number:"))

sum=sum+number
```

```
count+=1
print("Sum="+str(sum))
```

修改后代码：

```
#-*- coding:utf-8 -*-
count=1
sum=0
numberCount=int(input("How many numbers for summation :"))
while(1):
if count>numberCount:
break
else:
number = int(input("Enter a number:"))
sum=sum+number
count+=1
print("Sum="+str(sum))
```

实验三　嵌套循环结构程序设计

3.1　实验内容

1. 设计一个程序，输出一个提示信息，请求用户输入班级数量及学生数量，如果某个班级学生数量大于 50，则提示出错信息，重新进行输入，否则使用嵌套循环结构进行学生成绩统计，且需要在屏幕上给出计算结果。

2. 通过循环打印出如下图案：

输入 5:

打印出：

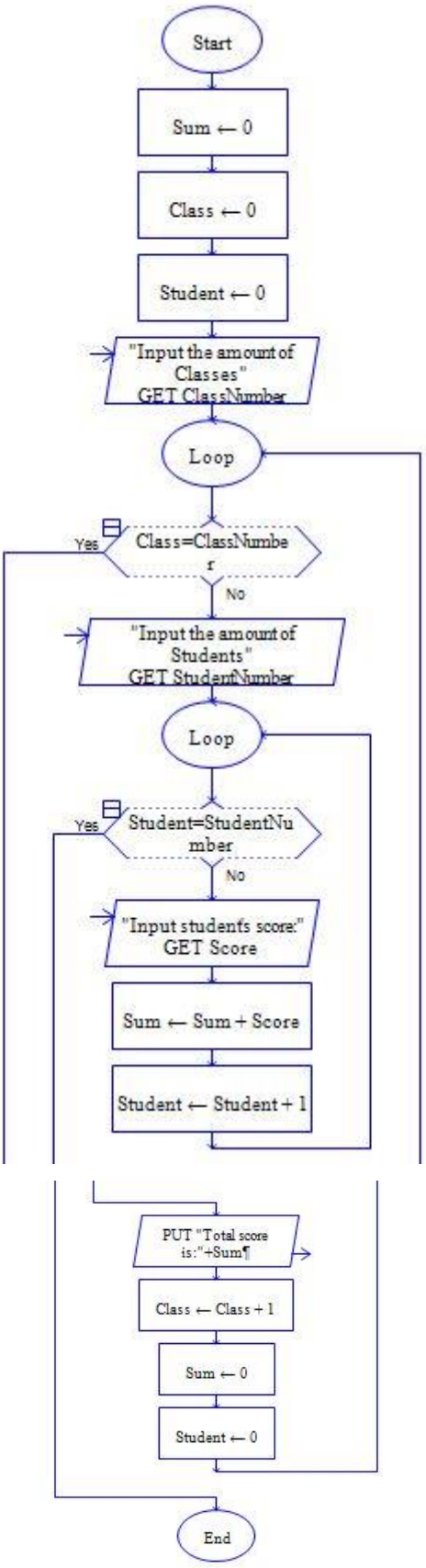
```
    *
   ***
  *****
```

3. 使用嵌套循环编写一个小型的猜数字游戏，内重循环要产生一个从 1 到 10 的随机数，可以允许用户猜 3 次，对每一个数字，显示出来该数字是大、小还是相等，猜中了或者 3 次机会用完，提示用户继续还是结束游戏。最后打印输出产生了几次随机数，猜对了几次。

3.2　实验目的与要求

了解嵌套循环结构的程序设计方法；编辑和运行 Python 程序。实验要求采用嵌套循环结构进行程序设计，给出程序设计流程图或伪代码，并使用 Python 代码编程实现，且编写实验报告。

3.3 RAPTOR 参考流程图



3.4 C++参考实现代码

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
stringraptor_prompt_variable_zzyz;
int score;
intclassnumber;
intstudentnumber;
int sum;
intclassnum;
int student;

sum =0;
classnum =0;
student =0;
raptor_prompt_variable_zzyz ="Input the amount of Classes";
cout<<raptor_prompt_variable_zzyz<<endl;
cin>>classnumber;
while (!(classnum==classnumber))
    {
raptor_prompt_variable_zzyz ="Input the amount of Students";
cout<<raptor_prompt_variable_zzyz<<endl;
cin>>studentnumber;
while (!(student==studentnumber))
        {
raptor_prompt_variable_zzyz ="Input student's score:";
cout<<raptor_prompt_variable_zzyz<<endl;
cin>> score;
sum =sum+score;
student =student+1;
        }
cout<< "Total score is:" << sum <<endl;          classnum =classnum+1;
sum =0;
student =0;
    }
return 0;
}
```

3.5 PYTHON 参考实现代码

```
# -*- coding:utf-8 -*-
sum=0
classnum=0
studentnum=0
classnumber=int(input("Input the amount of Classes:"))
while(classnum<classnumber):
ifclassnum==classnumber:
break
else:
studentnumber = int(input("Input the amount of Students:"))
while(studentnum<studentnumber):
ifstudentnum==studentnumber:
break
else:
score = int(input("Input student's score:"))
sum=sum+score;
studentnum+=1;
print("Total score is "+str(sum))
classnum+=1;
sum=0
studentnum=0
```

3.6 实验思考题

1. 什么样的问题会使用循环控制？
2. 如何避免出现死循环的情况？
3. 循环控制有哪些类型？

实验四　数组程序设计

4.1　实验内容

1.设计一个程序，已知一个班级 30 个学生的英语成绩，使用一维数组保存该班级学生的英语成绩，并分别统计 60 分以上及 90 分以上学生的数量且输出统计结果。

2.用程序实现两个字符串的比较、追加、拷贝

3. 两个具有相同行数和列数的矩阵可以相乘得出第三个矩阵。假设有如下两个矩阵：

$$A=\begin{bmatrix} a_{11}a_{12}.....a_{1n} \\ a_{21}a_{22}.....a_{2n} \\ ..... \\ a_{n1}a_{n2}.....a_{nn} \end{bmatrix} \qquad B=\begin{bmatrix} b_{11}b_{12}.....b_{1n} \\ b_{21}b_{22}.....b_{2n} \\ ..... \\ b_{n1}b_{n2}.....b_{nn} \end{bmatrix}$$

矩阵 A 和 B 的乘积是第三个矩阵 C，其大小为 n*n, C 的每个元素由下面等式给定：

$$C_{ij}=\sum_{k=1}^{n} a_{ik}b_{kj}$$

编写一个程序先读取 A 和 B 的元素，再计算矩阵 C。

② A、B 两个数组的值在程序中通过初始化方式得到。

② A、B 两个数组的值通过输入函数得到，并检查结果是否正确。

4. 请编写一个程序，从键盘读取一个字符串，并确定该字符串是否为回文。（顺读和倒读都一样的字符串就称为回文。例如：Madam 和 Anna 就是回文字符串。忽视大小写。）

至少准备两组测试数据：

①字符串是回文，例如：madam。

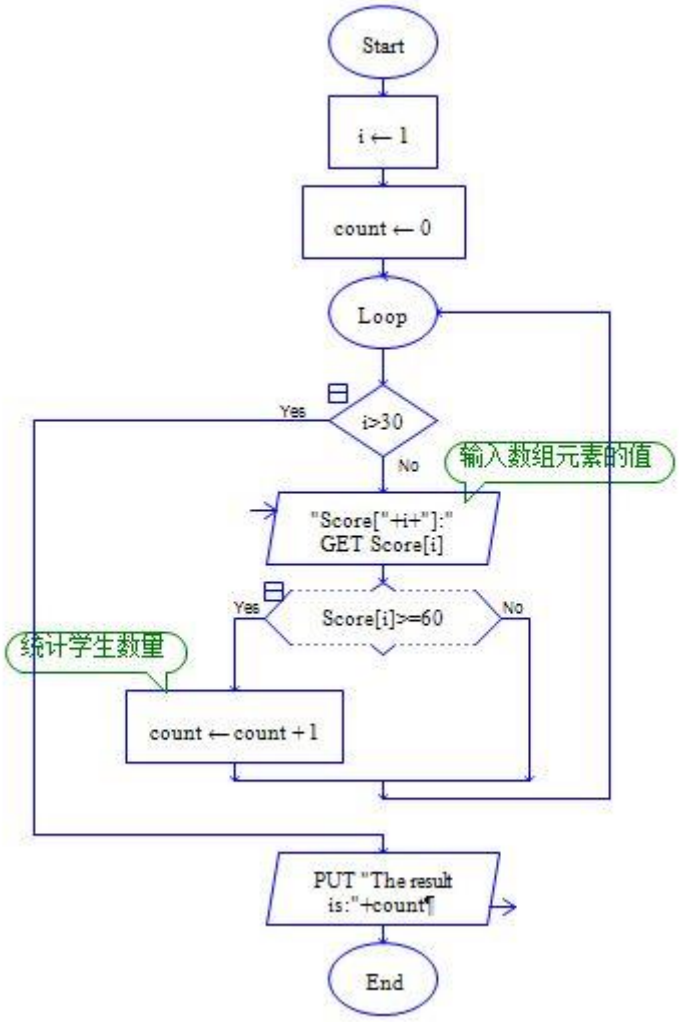
②字符串不是回文，例如：final。

4.2　实验目的与要求

掌握数组的程序设计方法；编辑和运行 Python 程序。实验要求采用数组类

型进行程序设计，给出程序设计流程图或伪代码，并使用 Python 代码编程实现，且编写实验报告。

4.3 RAPTOR 参考流程图



4.4 C++参考实现代码

```
#include <iostream>
#include <string>

using namespace std;
int main()
{
string raptor_prompt_variable_zzyz1;
string raptor_prompt_variable_zzyz2;
```

```
inti;
int count;
int score[30];

i =1;
count =0;
while (!(i>30))
    {
        raptor_prompt_variable_zzyz1 ="Score[";
        raptor_prompt_variable_zzyz2 ="]:";
cout<< raptor_prompt_variable_zzyz1<<i<< raptor_prompt_variable_zzyz2<<endl;
cin>> score[i];
if (score[i]>=60)
        {
count =count+1;
        }
else
        {
        }
i++;
    }
cout<< "The result is:" << count <<endl;
return 0;
}
```

4.5 PYTHON 参考实现代码

```
count=0

score=[0]*30

i=0

while(i<30):

    score[i]=int(input("Score["+str(i)+"]:"))

if score[i]>=60:

count+=1

i+=1

print("The result is :"+str(count))
```

4.6 实验思考题

1．列表有哪些灵活性？

2．使用列表有哪些注意事项？

实验五　查找与排序程序设计

5.1　实验内容

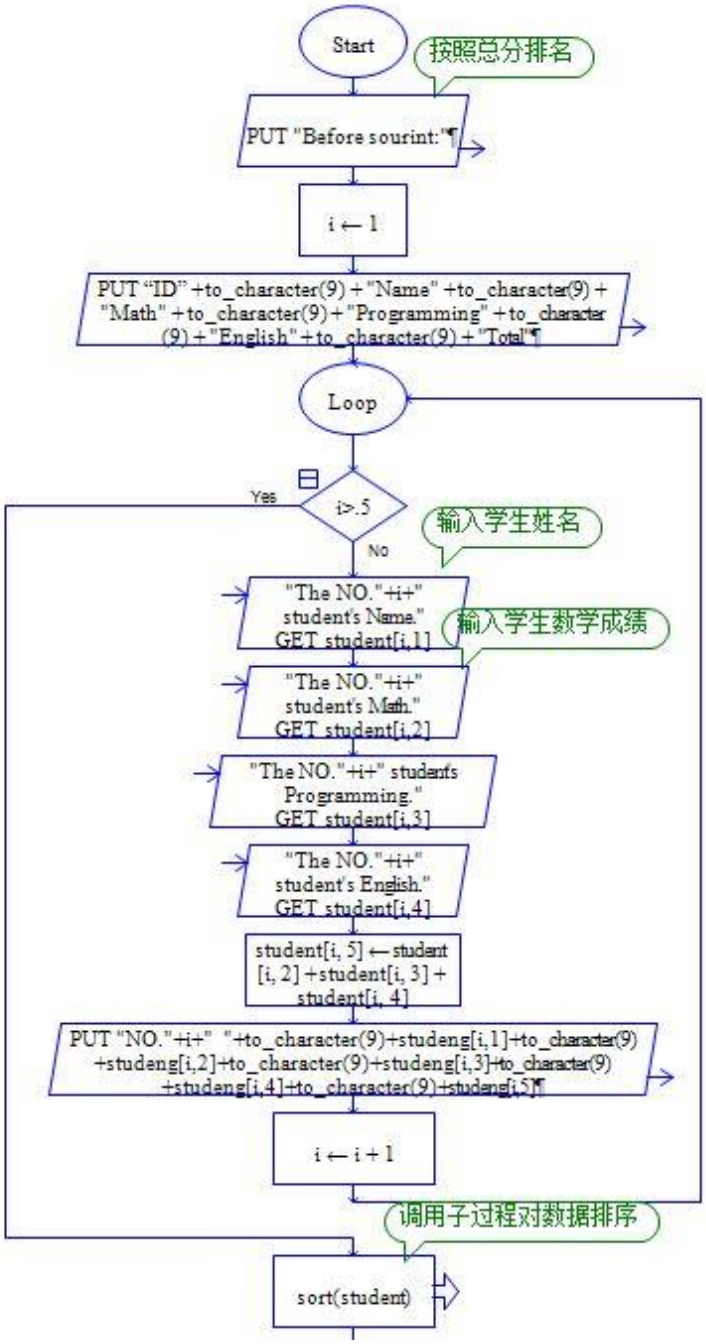
1. 设计一个程序，对学生总成绩进行排名。假设有 5 名学生成绩，包括姓名和 3 门课的成绩，计算每个学生的课程成绩总分以及其平均分，输出计算结果并按照总分成绩由高到低排名。

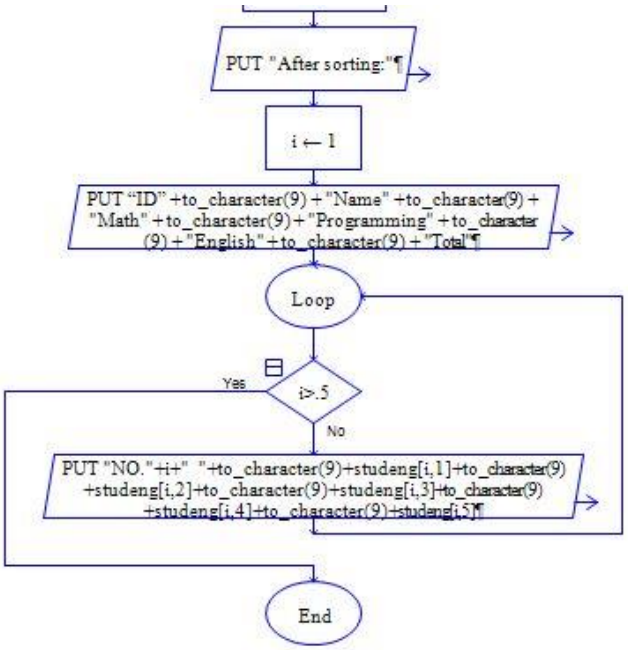
2. 用赋初值的方法将 15 个数存放在一个数组中，首先对这 15 个数进行排序；接着使用输入函数输入一个数，要求用折半查找法找出该数是数组第几个元素的值。如果该数不在数组中，则输出“无此数”。

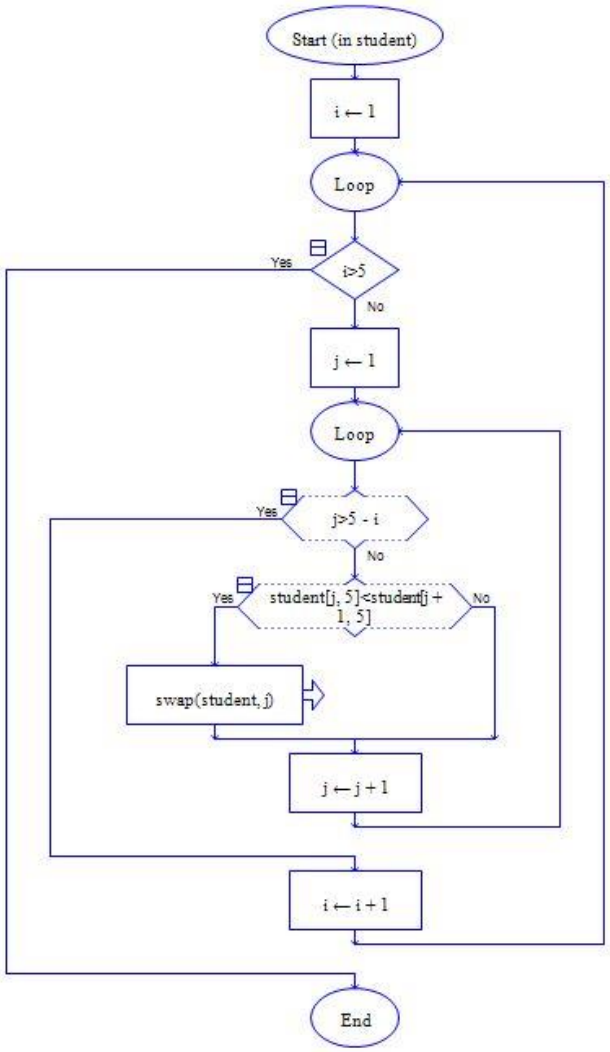
5.2　实验目的与要求

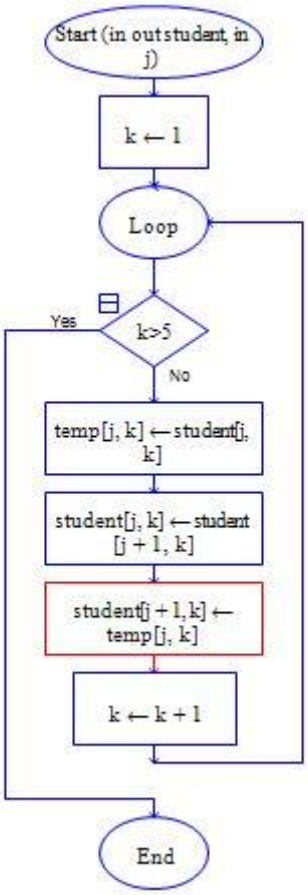
掌握查找与排序算法；编辑和运行 Python 程序。实验要求采用查找与排序算法进行程序设计，给出程序设计流程图或伪代码，并使用 Python 代码编程实现，且编写实验报告。

5.3 RAPTOR 参考流程图




PUT "After sorting:"¶
i ← 1
PUT "ID" +to_character(9) + "Name" +to_character(9) + "Math" +to_character(9) + "Programming" +to_character (9) + "English" +to_character(9) + "Total"¶
Loop
Yes
i>.5
No
PUT "NO."+i+" "+to_character(9)+studeng[i,1]+to_character(9) +studeng[i,2]+to_character(9)+studeng[i,3]+to_character(9) +studeng[i,4]+to_character(9)+studeng[i,5]¶
End


Start (in student)
i ← 1
Loop
Yes
i>5
No
j ← 1
Loop
Yes
j>5 - i
No
Yes
student[j, 5]<student[j + 1, 5]
No
swap(student, j)
j ← j + 1
i ← i + 1
End



5.4 PYTHON 参考实现代码

```
# -*- coding:utf-8 -*-
importnumpy as np
def swap(student,j):
    k=0
while(k<6):
temp=student[j][k]
student[j][k]=student[j+1][k]
student[j+1][k]=temp
        k+=1
def sort(student):
i=0
while(i<5):
```

```
            j=0
while(j<5-i-1):
ifint(student[j][5])<int(student[j+1][5]):
swap(student,j)
                    j+=1
i+=1
print("Before sorting:")
#print("ID、Name、Math、Programming、English、Total")
i=0
students=np.zeros((5,6),object)
print(students)
while(i<5):
students[i][0]=i;
name=input("The NO "+str(i)+" student's name :")
students[i][1]=name;
math=input("The NO "+str(i)+" student's math :")
students[i][2]=math;
programming=input("The NO "+str(i)+" student's programming :")
students[i][3]=str(programming);
english=input("The NO "+str(i)+" student's english :")
students[i][4]=english;
total=int(math)+int(programming)+int(english)
students[i][5]=str(total)
print(students)
      print("ID:" + str(students[i][0]) + " Name:" + str(students[i][1]) + " Math:" + str(
            students[i][2]) + " Programming:" + str(students[i][3]) + " English:" +
str(students[i][4]) + " Total:" + str(
students[i][5]))
i += 1
sort(students)
```

```
print("After sorting:")
i=0
while(i<5):
    print("ID:" + str(students[i][0]) + " Name:" + str(students[i][1]) + " Math:" + str(
        students[i][2]) + " Programming:" + str(students[i][3]) + " English:" +
str(students[i][4]) + " Total:" + str(
students[i][5]))
i += 1
```

5.5 实验思考题

1．如何提高查找与排序的执行效率？

2．排序算法通常使用哪种数据结构？

实验六　模块化程序设计

6.1　实验内容

1. 设计一个程序，输入一组正数，存储到数组中，求出这组数的平均值并统计这组数中大于等于平均值的数量，输出计算结果。程序使用不同模块，分别完成计算平均值、判断符合条件数量、输出计算结果。

2. 求两个数的最大公约数，这两个数通过输入函数输入。

3. 编写一个菜单驱动式程序，输入一个数值 X，按照用户的选择，输出面积 Area 的值：

正方形的边长为 X，A=X*X

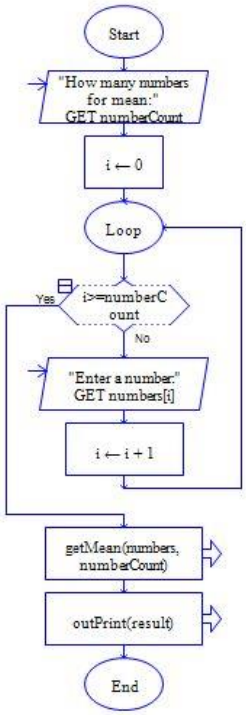
圆形的半径为 X，A=3.14*X*X

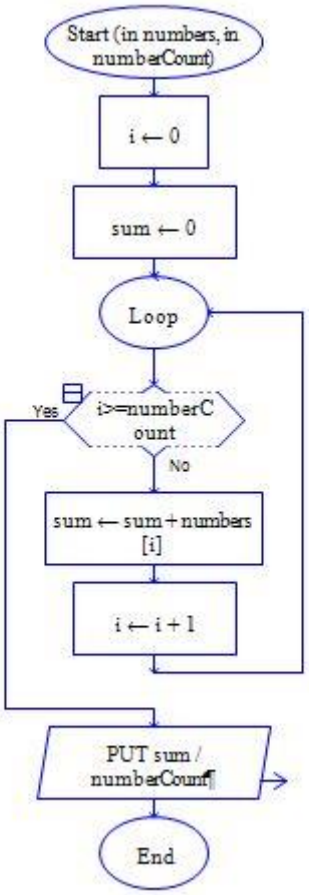
等边三角形的边长为 X，A=Sqrt（3）*X*X/4

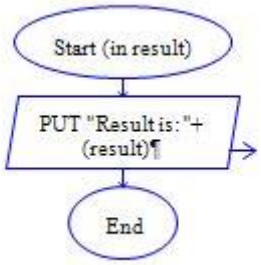
6.2　实验目的与要求

掌握模块化程序设计方法；编辑和运行 Python 程序。实验要求采用模块化程序设计方法，给出程序设计流程图或伪代码，并使用 Python 代码编程实现，且编写实验报告。

6.3 RAPTOR 参考流程图







6.4 PYTHON 参考实现代码

```
# -*- coding:utf-8 -*-
importnumpy as np
defgetMean(numbers,numberCount):
i=0
sum=0
while(i<numberCount):
sum+=numbers[i]
i+=1
return sum/numberCount
defoutPrint(result):
print("Result is :" + str(result))
numberCount=int(input("How many numbers for mean :"))
numbers = np.zeros(numberCount,int)
i=0
while(i<numberCount):
numbers[i] = int(input("Enter a number:"))
i+=1
result=getMean(numbers,numberCount)
outPrint(result)
```

6.5 实验思考题

1．模块化程序设计的特点？

2．模块是如何划分的？

实验七　数据文件程序设计

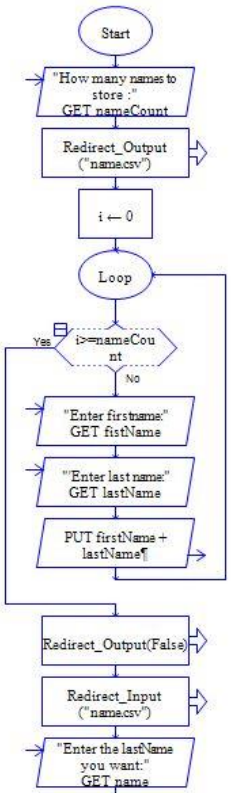
7.1　实验内容

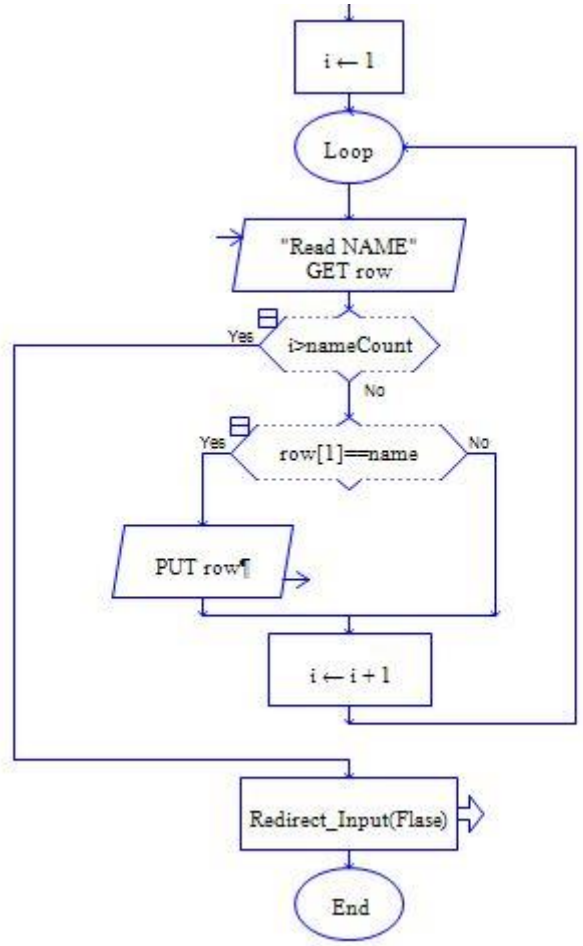
设计一个程序，保存一个客户文件，记录了客户姓名，用户输入姓之后，程序能够搜索文件，并输出所有该姓氏的客户姓名，计算同姓客户的数量，输出计算结果。

7.2　实验目的与要求

了解文件操作的程序设计方法；编辑和运行 Python 程序。实验要求采用数据文件进行程序设计，学会使用文件打开、关闭、读、写等文件操作，给出程序设计流程图或伪代码，并使用 Python 代码编程实现，且编写实验报告。

7.3　RAPTOR 参考流程图





7.4 PYTHON 参考实现代码

```
# -*- coding:utf-8 -*-
import csv
nameCount=int(input("How many names to store :"))
i=0
while (i<nameCount):
with open('names.csv', 'a') as nameFile:
        Writer = csv.writer(nameFile)
firstName = input("Enter name["+str(i)+"]'s firstName :")
lastName = input("Enter name["+str(i)+"]'s lastName :")
Writer.writerow([firstName,lastName])
nameFile.close()
i+=1
```

```
csv_reader = csv.reader(open('names.csv', encoding='utf-8'))
name=input("Enter the lastName you want :")
for row in csv_reader:
if row[1]==name:
print(row)
```

实验八　面向对象程序设计

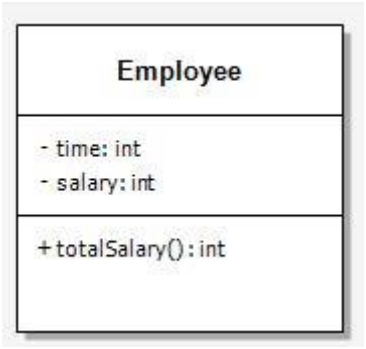
8.1　实验内容

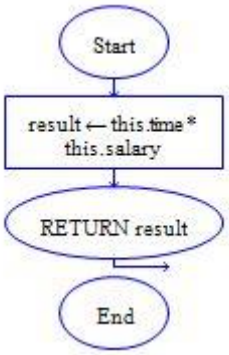
设计一个程序，由用户输入雇员的数量、工时数及每小时的工资，程序采用面向对象的方法，输出雇员的总工资并对超出总工资平均值的雇员给予证书奖励。

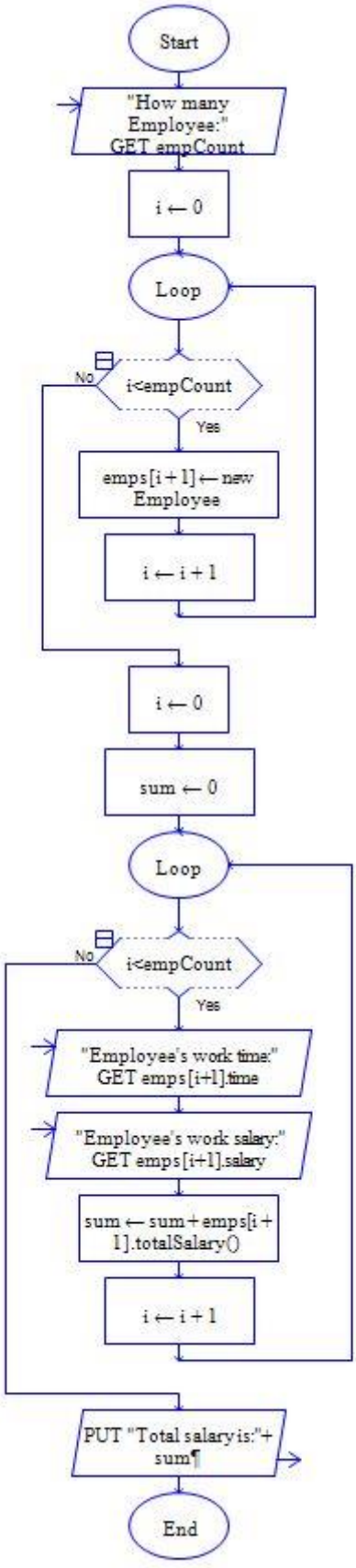
8.2　实验目的与要求

掌握面向对象程序设计思想，掌握类图的绘制方法，编辑和运行 Python 程序。实验要求采用面向对象程序设计方法，给出程序设计流程图或伪代码，并使用 Python 代码编程实现，且编写实验报告。

8.3　RAPTOR 参考流程图






Start
"How many Employee:" GET empCount
i ← 0
Loop
No
i<empCount
Yes
emps[i + 1] ← new Employee
i ← i + 1
i ← 0
sum ← 0
Loop
No
i<empCount
Yes
"Employee's work time:" GET emps[i+1].time
"Employee's work salary:" GET emps[i+1].salary
sum ← sum + emps[i + 1].totalSalary()
i ← i + 1
PUT "Total salary is:"+ sum¶
End

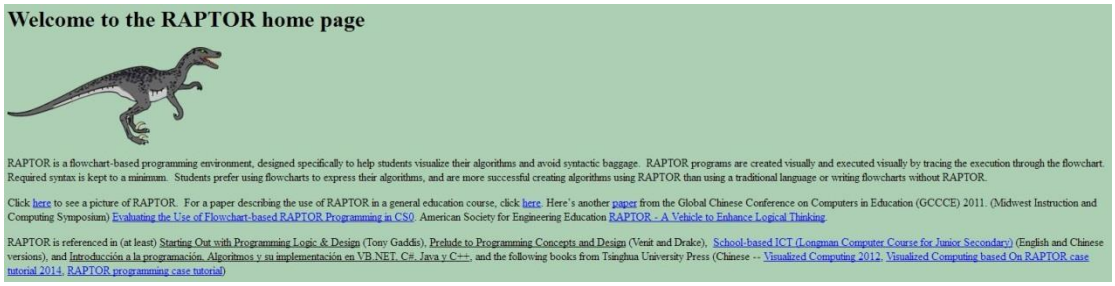
8.4 PYTHON 参考实现代码

```
# -*- coding: UTF-8 -*-
importnumpy as np
class Employee:
empCount=0
def __init__(self, time, salary):
self.time = time
self.salary = salary
deftotalSalary(self):
returnself.salary*self.time
defsetTime(self,time):
self.time=time
defsetSalary(self,salary):
self.salary=salary

Employee.empCount=int(input("How many Employees :"))
emps=np.zeros(Employee.empCount,Employee)
i=0
while(i<Employee.empCount):
emps[i]=Employee(0,0)
i+=1
i=0
sum=0
while(i<Employee.empCount):
emps[i].setTime(int(input("NO "+str(i)+" Employee's work time:")))
emps[i].setSalary(int(input("NO "+str(i)+" Employee's work salary:")))
sum = sum + emps[i].totalSalary()
i+=1
print("Total salary is :"+str(sum))
```

附录一　RAPTOR 程序设计环境简介

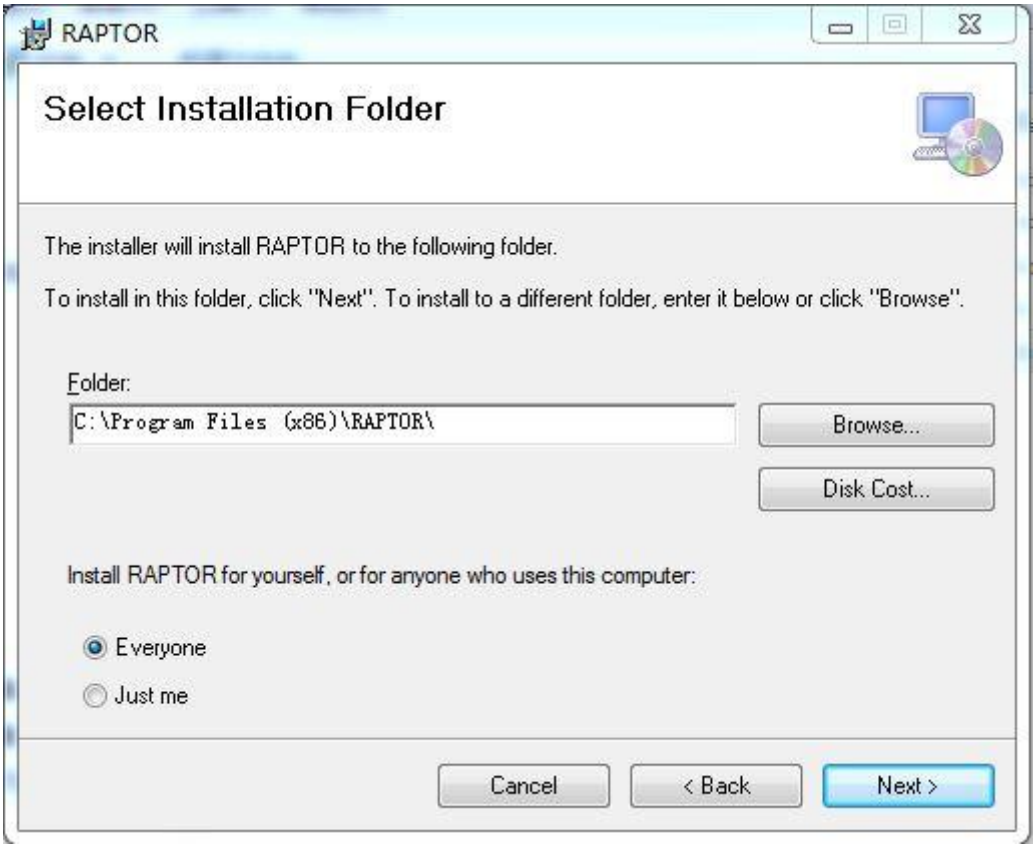
RAPTOR 安装过程：
访问如下网址：http://raptor.martincarlisle.com/



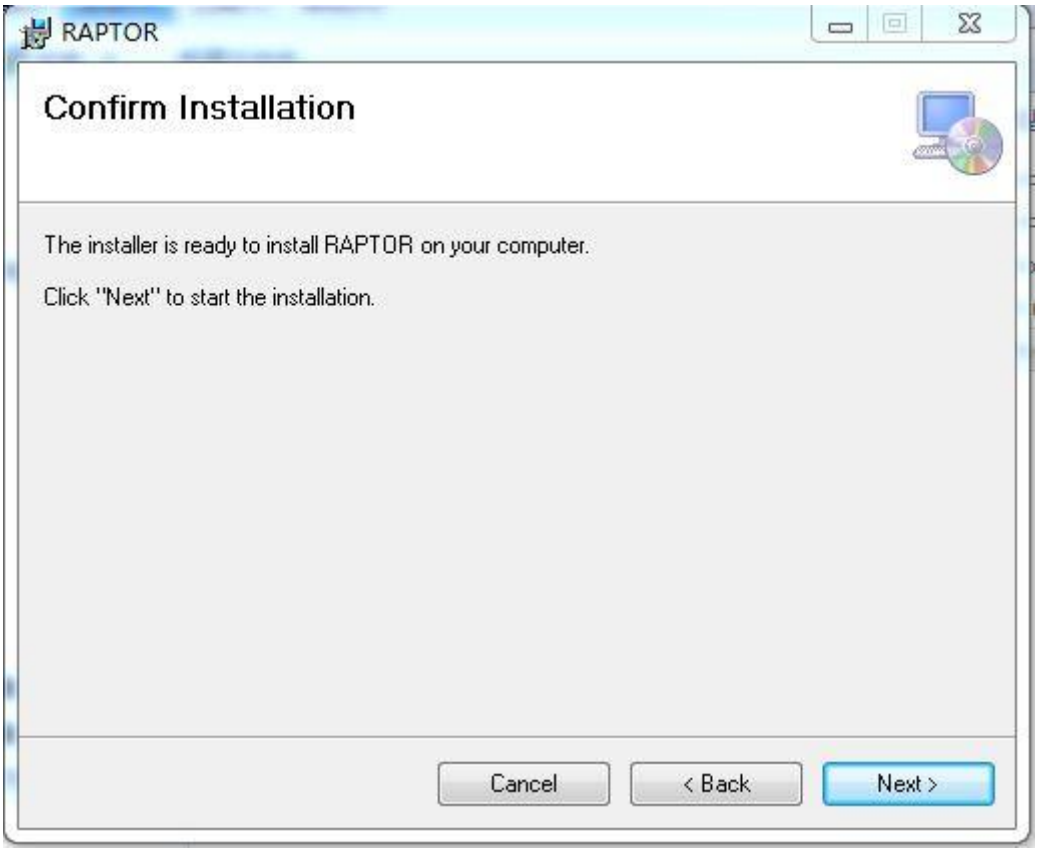
选择最新的 RAPTOR 版本，点击 Download latest version 按钮：

例如：2016 版本，文件名称为：raptor2016，类型为 Windows Installer 程序包，双击该程序安装包后出现：

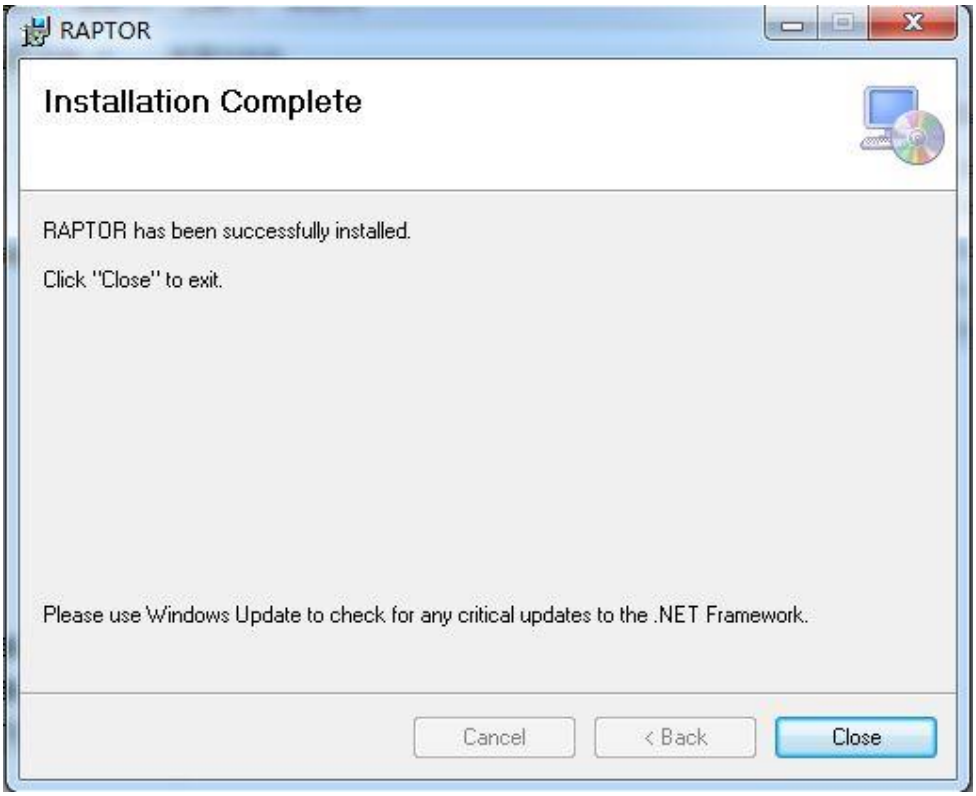
点击 Next 按钮之后出现：



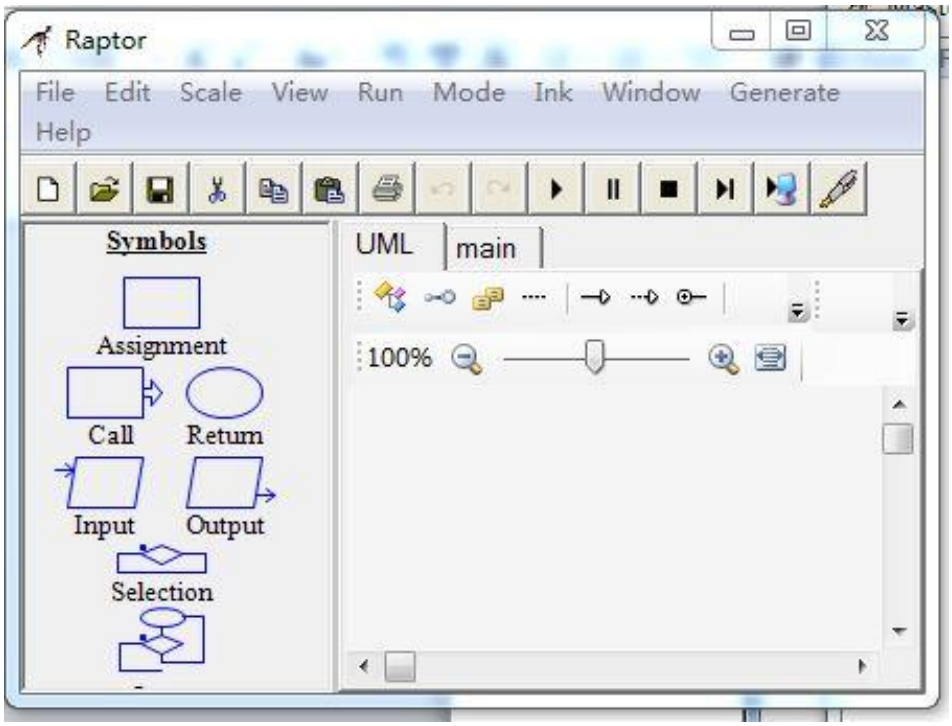
继续点击 Next 按钮之后出现：



继续点击 Next 按钮之后开始执行安装过程，完成之后将出现：



点击 Close 按钮安装过程结束。可以通过启动“开始菜单”中的 RAPTOR 来运行，出现：

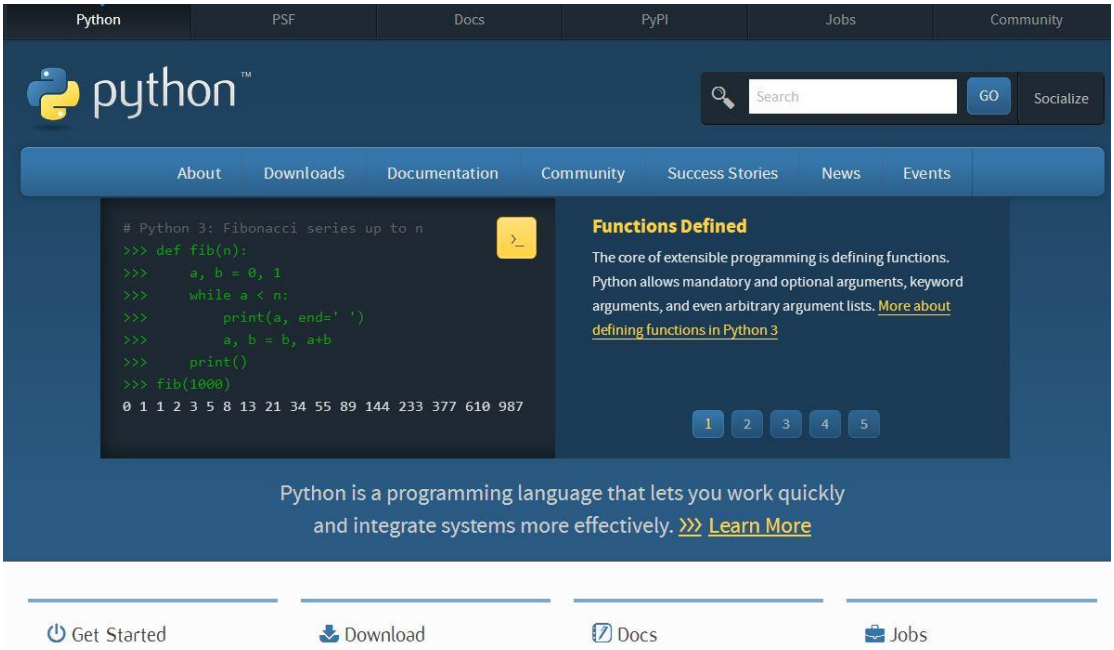


表明安装成功。

附录二　Python 安装简介

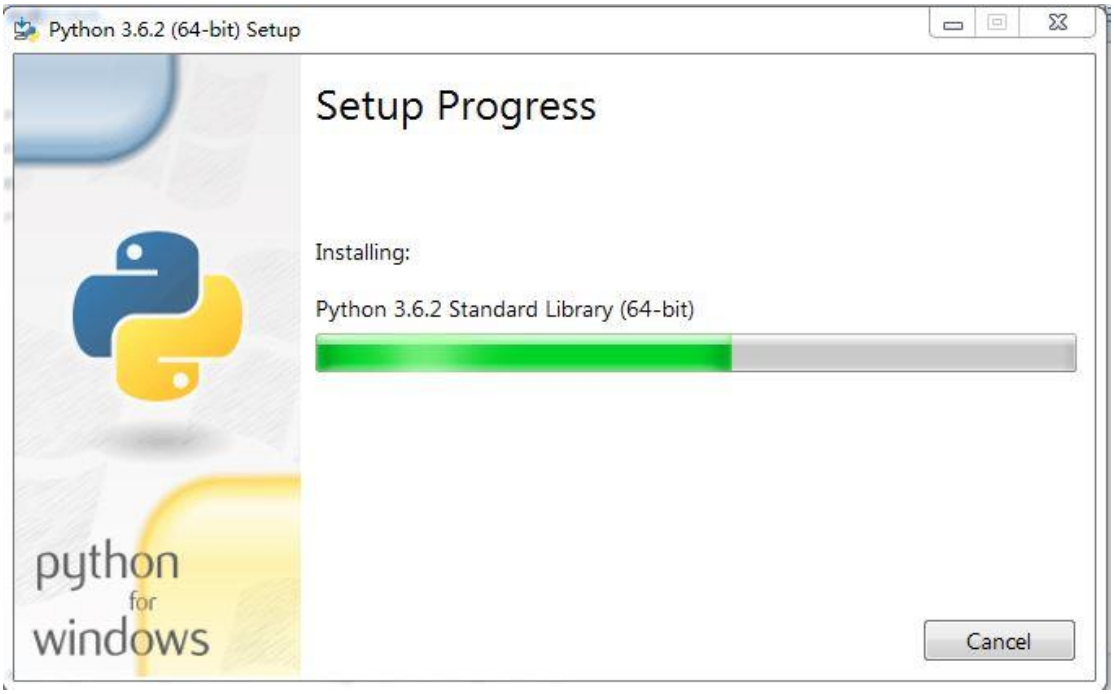
Python 安装过程

访问如下网址：https://www.python.org

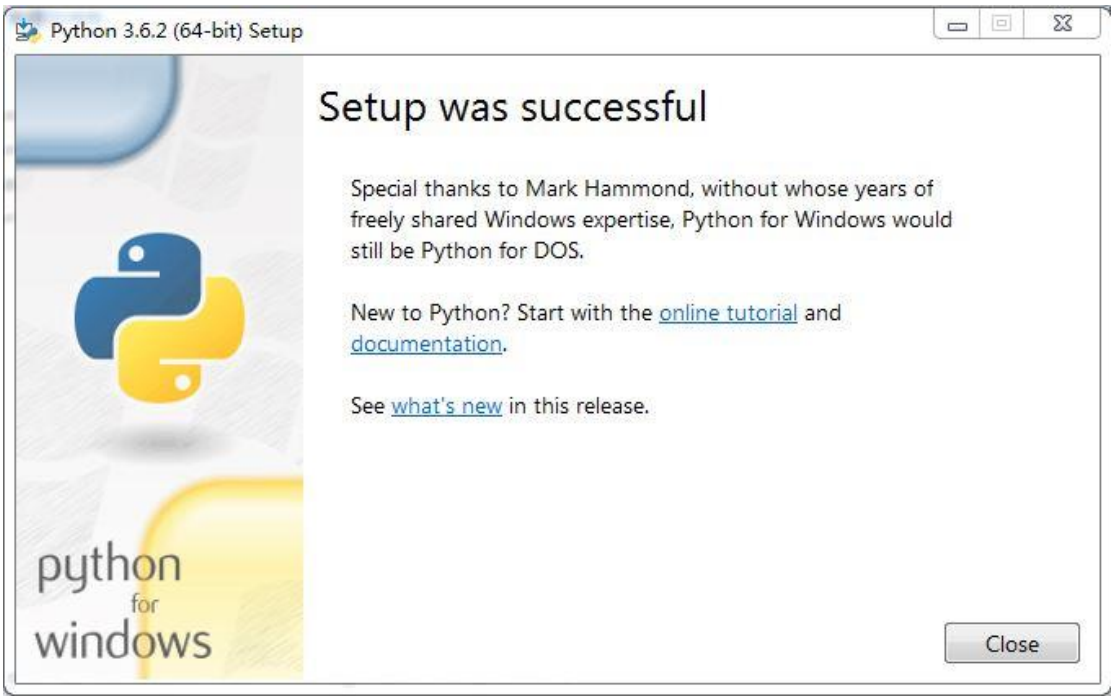


选择需要下载的 Python 版本，例如：3.6.2，文件名称为：python-3.6.2.amd64，类型为应用程序安装包，双击该程序安装包后出现：

勾选 Add Python 3.6 to PATH，然后选择 Install Now 开始安装过程，将出现：

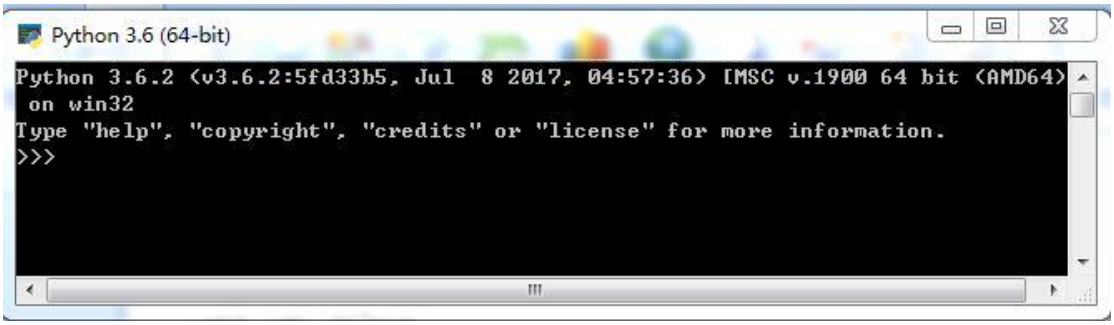


然后出现：

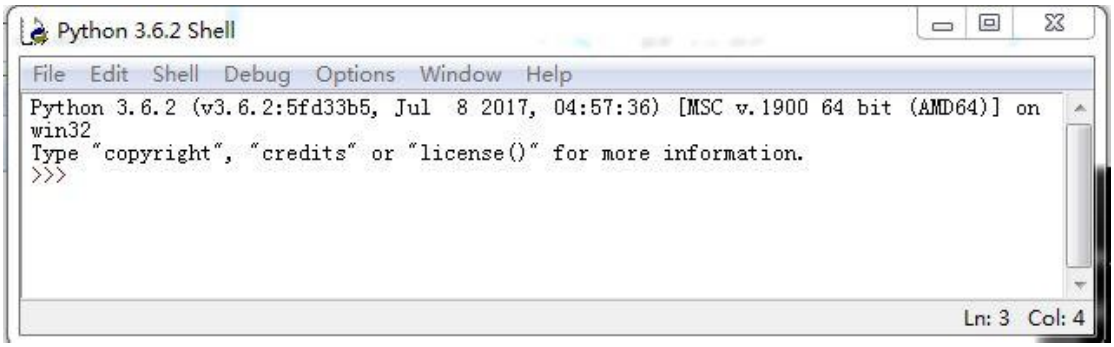


点击 Close 按钮之后安装完成。

可以通过启动“开始菜单”中的 Python3.6 来运行 Python 环境表明安装成功。如下所示：



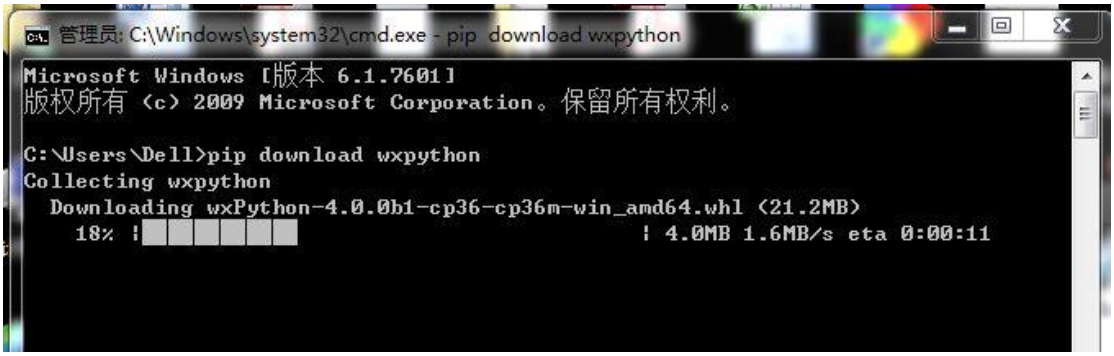
或者如下所示：



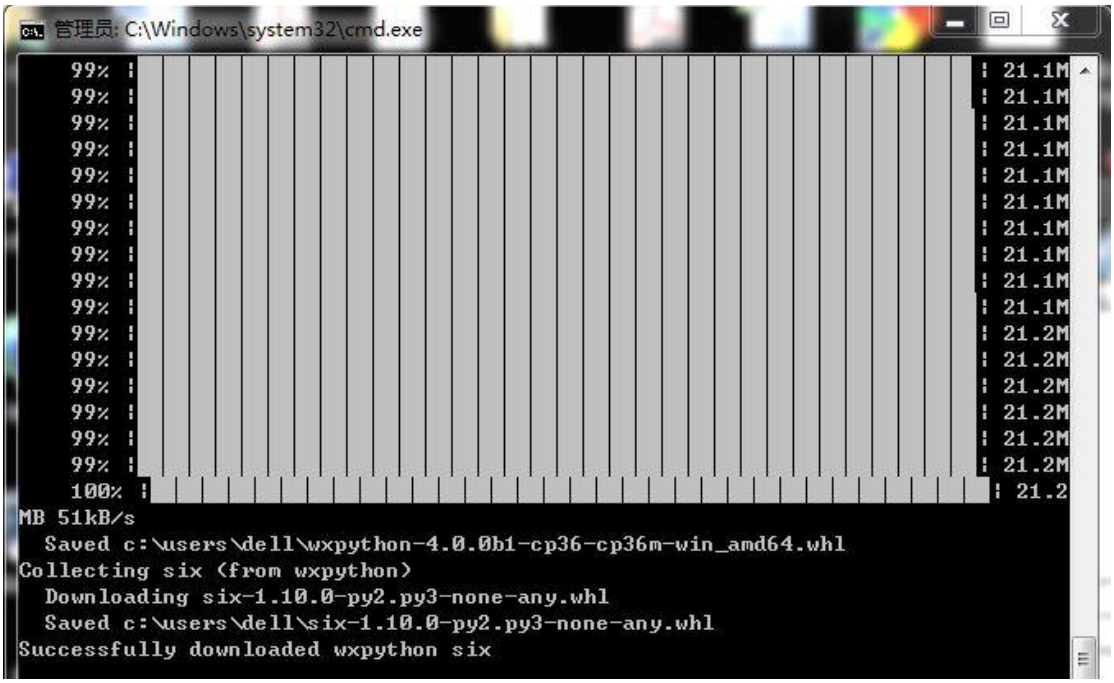
可以开始编写 Python 程序。此外，为了编写图形界面程序，需要下载一个 GUI 安装包 wxpython。下载及安装过程如下：

在命令行下输入 pip download wxpython 命令，如下所示：

按回车之后出现：



等待下载 wxpython，全部下载完成之后出现：



在命令行下输入 pip install wxpython 命令并按回车键开始 wxpython 工具包的安装过程，安装命令及安装完成之后如下所示：

```
C:\Users\Dell>pip install wxpython
Collecting wxpython
  Using cached wxPython-4.0.0b1-cp36-cp36m-win_amd64.whl
Collecting six (from wxpython)
  Using cached six-1.10.0-py2.py3-none-any.whl
Installing collected packages: six, wxpython
Successfully installed six-1.10.0 wxpython-4.0.0b1
```

附录三　程序编码风格

编码风格决定了程序代码的可读性，好的编程风格包括结构清晰、条理性强、字体工整、缩进合理，质量高的程序代码需要遵循书写规范。

1，注释

添加注释不仅有助于编程者日后阅读，也方便其他维护人员理解代码的含义。通常包括作者、版权、程序功能、关键代码等方面的注释说明。

2，空行

编译器不会对空行处理，仅仅起到分隔代码段落、区分层次的作用，因此可以使用空行分隔逻辑不相关的代码模块。

3，代码行

代码行尽量一行代码只做一件事情，这样方便进行调试。

4，空格

在适当的地方添加空格，有利于代码的阅读。

5，缩进与对齐

在代码中适当进行缩进与对齐，既是某些语法规定又是便于阅读的方式。

6，命名规则

如何对变量和函数命名非常重要，高质量代码需要合理的命名规则，可以让阅读者望文知意，比较公认的命名规则是匈牙利命名法，即在函数名或变量名之前加上前缀以说明其含义，增进阅读者的理解。

附录四　实验报告模板

课程编号：C0800000012

程序设计基础实验报告

姓　　名		学 号	
班　　级		指 导 教 师	
实 验 名 称	程序设计基础实验		
开 设 学 期	2017-2018 第 一 学 期		
开 设 时 间	第 8 周 — — 第 17 周		
报 告 日 期			
评 定 成 绩		评　　定　　人	
		评 定 日 期	

东北大学软件学院

实验名称（如：实验二 循环结构程序设计）

一、实验目的

说明本次实验所涉及并要求掌握的知识点。参见实验指导书中对实验目的的描述。

二、问题分析与程序设计

通过对实验题目中给出的问题的详细分析，绘制程序设计的流程图或者写出伪代码，并对流程图进行相关的注释。

三、实现过程与测试结果分析

针对程序设计流程，给出 Python 实现代码，列出调试过程中出现的问题、调试的过程和解决的方法。

四、实验结果总结

对实验结果进行分析，列出测试数据以及相应输出结果，给出程序设计流程图和关键运行界面的截屏。

五、创新的部分

如果有创新的内容，在此写明。

六、对实验的意见与建议

总结实验的心得体会，并提出对实验的改进意见。

评价表格（每份实验报告只需一份评分表）

考核标准	得分
（1）正确理解和掌握实验所涉及的概念和原理（20%）；	
（2）按实验要求合理设计程序执行流程（20%）；	
（3）能编程实现设计的程序流程，运行结果正确（20%）；	
（4）认真记录实验数据，原理及实验结果分析准确（20%）；	
(5)实验过程中,具有严谨的学习态度和认真、踏实、一丝不苟的科学作风(5%)；	
（6）所做实验具有一定的创新性（5%）；	
（7）实验报告规范（10%）。	