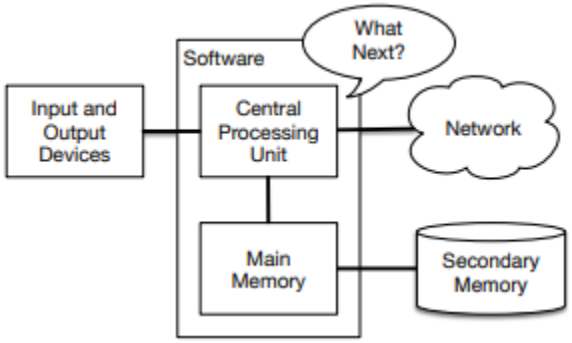
8/26 CSP P4E Ch.1 Notes:

8/26 CSP P4E Ch.1 Notes:

- Computers are always wanting to do what's next(1)
- Traditionally the boring tasks for humans and the ones computers excel in(1)
-
- Once you learn this new language, you can delegate mundane tasks to your partner (the computer), leaving more time for you to do the things that you are uniquely suited for.(2)
- Personal Digital Assistant (PDA) (2)



-
- The Central Processing Unit (or CPU) is the part of the computer that is built to be obsessed with “what is next?”(3)
- The Main Memory is where information is stored that the CPU will need fast.
- The Second Memory also stores information, but slower then main. However it can store information even with no power to the computer. For example: USB sticks, portable music player
- The Input and Output Devices are simply our screen, keyboard, mouse, microphone, speaker, touchpad, etc. It is basically all of the ways we interact with
- the computer.
- Network connections are used by computers in order to retrieve information over a network. It is a very slow place to store and retrieve data that might not always be “up”. The network is a slower and at times unreliable form of Secondary Memory.
- As a programmer you will mostly be “talking” to the CPU and telling it what to do next. Sometimes you will tell the CPU to use the main memory, secondary memory, network, or the input/output devices.(4)
- ,you must write down your instructions in advance. It’s called stored instructions; a program and the act of writing these instructions down and getting the instructions to be correct programming.

- you need to “tell a story”. In writing a story, you combine words and sentences to convey an idea to the reader

The reserved words in the language where humans talk to Python include the following:

```
False      await      else       import     pass
None       break      except     in         raise
True       class      finally    is         return
and        continue   for        lambda     try
as         def        from       nonlocal   while
assert     del        global     not        with
async      elif       if         or         yield
```

```
print('Hello world!')
```

```
Hello world!
```

-
- type python and the Python interpreter will start executing in interactive mode
- Type “quit()” to say goodbye
- The CPU understands a language we call machine language. It consists of a bunch of 0, 1
- No one uses it though, they use translators so they can use powerful languages instead. Since machine language is tied to the computer hardware, machine language is not portable across different types of hardware.
- 2 types of translators: interpreters and compilers
- An interpreter reads the source code of the program as written by the programmer, parses the source code, and interprets the instructions on the fly.
- Python is an interpreter
- variable to refer to the labels we use to refer to this stored data.

```
>>> x = 6
>>> print(x)
6
>>> y = x * 7
```

-
- A compiler needs to be handed the entire program in a file, and then it runs a process to translate the high-level source code into machine language and then the compiler puts the resulting machine language into a file for later execution.
- No need for quit at end of python program in the file because python knows when it reaches end of file.
- Program is a sequence of python statements that have been crafted to do something.

```
(base) C:\Users\colin>python hello.py
Enter file: test.txt
She 3
```

-

- input Get data from the “outside world”. This might be reading data from a file, or even some kind of sensor like a microphone or GPS. In our initial programs, our input will come from the user typing data on the keyboard.
- output Display the results of the program on a screen or store them in a file or perhaps write them to a device like a speaker to play music or speak text.
- sequential execution Perform statements one after another in the order they are encountered in the script.
- conditional execution Check for certain conditions and then execute or skip a sequence of statements.
- repeated execution Perform some set of statements repeatedly, usually with some variation.
- reuse Write a set of instructions once and give them a name and then reuse those instructions as needed throughout your program.
- Type of errors: Syntax, Logic, Semantic
- Syntax: violatred grammar rules and usually easy to fix
- Logic: u have good syntax but the order of the statements are out of order or a mistake with how the statements relate to each other
- Semantic: A semantic error is when your description of the steps to take is syntactically perfect and in the right order, but there is simply a mistake in the program. The program is perfectly correct but it does not do what you intended for it to do
- 4 ways to debug: reading, running, ruminating, retreating
- Reading: re read the code and make sure it says what u meant for it to say
- Running: make changes and run different versions.
- Ruminating: take some time to think about it, like what type of error is it.
- Retreating :At some point, the best thing to do is back off, undoing recent changes, until you get back to a program that works and that you understand. Then you can start rebuilding.
- You have to take time to think. Debugging is like an experimental science. You should have at least one hypothesis about what the problem is. If there are two or more possibilities, try to think of a test that would eliminate one of them.
- Sometimes the best option is to retreat, simplifying the program until you get to something that works and that you understand.
-

Exercise 1: What is the function of the secondary memory in a computer?
a) Execute all of the computation and logic of the program
b) Retrieve web pages over the Internet
c) Store information for the long term, even beyond a power cycle
d) Take input from the user
Exercise 2: What is a program?

A program is sequence of python statements that have been crafted to do something

Exercise 3: What is the difference between a compiler and an interpreter?
A interpreter takes the source code and analyzes and interprets it. A compiler needs the whole program and then translates it into machine language.
Exercise 4: Which of the following contains “machine code”?
a) The Python interpreter

b) The keyboard
c) Python source file
d) A word processing document

Exercise 5: What is wrong with the following code:
>>> primt 'Hello world!'
File "<stdin>", line 1
primt 'Hello world!'
^
SyntaxError: invalid syntax

Misspelt print and didnt include parentheses

Exercise 6:Where in the computer is a variable such as “x” stored after the following Python line finishes?
x = 123
a) Central processing unit
b) Main Memory
c) Secondary Memory
d) Input Devices
e) Output Devices

Exercise 7: What will the following program print out:
x = 43
x = x - 1
print(x)
a) 43
b) 42
c) x + 1
d) Error because x = x + 1 is not possible mathematically

Exercise 8: Explain each of the following using an example of a human capability: (1) Central processing unit, (2) Main Memory, (3) Secondary Memory, (4) Input Device, and (5) Output Device. For example, “What is the human equivalent to a Central Processing Unit”?