PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE

ACADEMIC YEAR: 2023-24

DEPARTMENT of COMPUTER ENGINEERING DEPARTMENT

CLASS: T.E. **SEMESTER: I**

SUBJECT: LP-1 (SP&OS)

ASSINGMENT NO.	A1 (Part 1)
TITLE	Implement pass-I of two-pass assembler.
PROBLEM STATEMENT/ DEFINITION	Design suitable data structures and implement pass-I of a two-pass assembler for pseudo-machine in Java. Implementation should consist of a few instructions from each category and few assembler directives.
OBJECTIVE	1. To learn systems programming tools 2. Implement language translator (i.e., assembler: a language translator for low level language like assembly language)
OUTCOME	After implementing this assignment, learners will be able to 1. Generate intermediate data structures like IC, SYMTAB, LITTAB, POOLTAB 2. Understand the relevance of the intermediate data structures with respect to Pass-II of assembler
S/W PACKAGES AND HARDWARE/ APPARATUS USED	**Operating System recommended:** 64-bit Open-source Linux or its derivative **Programming tools recommended:** Eclipse IDE **Programming language:** JAVA **Hardware requirements:** I3 and I5 machines
REFERENCES	Dhamdhere D., "Systems Programming and Operating Systems", McGraw Hill, ISBN 0 - 07 - 463579 – 4. Find eBook in the following link: https://drive.google.com/drive/folders/1vbu_x7DPHPaIKAbIngKquh5zgYtOgpTe?usp=sharing
STEPS	**Step1. Input for Pass-I of assembler:** 1. Assembly language program in hypothetical language as per the Author Dhamdhere, in a file. 2. OPTAB 3. Condition code table 4. Register table **Step2. Open the input file and apply the Pass1 of assembler algorithm, process the input file line by line and inside a line token by token** Please refer algorithm of Pass I of assembler from the following textbook: Dhamdhere D., "Systems Programming and Operating Systems", McGraw Hill, ISBN 0 - 07 - 463579 – 4. Please refer the following link for e-Book and solved test cases

	https://drive.google.com/drive/folders/1vbu_x7DPHPaIKAbIngKquh5zgYtOgpTe?usp=sharing **Step3. Output of Pass-I of assembler:** 1. IC 2. SYMTAB 3. LITTAB 4. POOLTAB
INSTRUCTIONS FOR WRITING JOURNAL	1. Date 2. Assignment no. 3. Problem definition 4. Learning objective 5. Learning Outcome 6. Concepts related Theory 7. Algorithm 8. Test cases 9. Conclusion/Analysis

Prerequisites:

Microprocessor, Data structures and Algorithm, Programming and problem solving

Concepts related Theory:

1. Assembler is a low-level translator who translates source code to machine code.
2. It works in two phases: analysis phase and synthesis phase.
3. In analysis phase, source assembly code is analyzed to generate some intermediate data structures.
4. In synthesis phase, machine code is generated.
5. There are well defined system level standard algorithms to design the assembler as a two-pass assembly, namely Pass-I and Pass-II algorithms of assembler.
6. Pass-I takes the source code in assembly language as input and generates intermediate data structures.

Algorithm:

Please refer algorithms of Pass-I of assembler from the following textbook: Dhamdhere D., "Systems Programming and Operating Systems", McGraw Hill, ISBN 0 - 07 - 463579 – 4.

Please refer the following link for algorithm in e-Book:

https://drive.google.com/drive/folders/1vbu_x7DPHPaIKAbIngKquh5zgYtOgpTe?usp=sharing

Conclusion:

Input assembly language program is processed by applying Pass-I algorithm of assembler and intermediate data structures, SYMTAB, LITTAB, POOLTAB, IC, are generated.

Review Questions:

1. What is two pass assembler?
2. What is the significance of symbol table?
3. Explain the assembler directives EQU, ORIGIN.
4. Explain the assembler directives START, END, LTORG.
5. What is the use of POOLTAB and LITTAB?
6. How are literals handled in Pass-I?
7. What are the tasks done in Pass-I?
8. How is error handling done in Pass-I?
9. Which variant is used in implementation? Why?
10. Which intermediate data structures are designed and implemented in Pass-I?

ASSINGMENT NO.	A1 (Part 2)
TITLE	Implement pass-II of two-pass assembler.
PROBLEM STATEMENT/ DEFINITION	Design suitable data structures and implement pass-II of a two-pass assembler for pseudo-machine in Java. Implementation should consist of a few instructions from each category and few assembler directives. The output of Pass-I (intermediate code file and symbol table) should be input for pass-II.
OBJECTIVE	1. To learn systems programming tools 2. Implement language translator (i.e., assembler: a language translator for low level language like assembly language)
OUTCOME	After implementing this assignment, learners will be able to 1. generate the machine code from intermediate data structures like IC, SYMTAB, LITTAB, POOLTAB understand the complete process of translator i.e translating a source program into the target program w.r.t low level translator i.e., assembler.
S/W PACKAGES AND HARDWARE/ APPARATUS USED	**Operating System recommended:** 64-bit Open-source Linux or its derivative **Programming tools recommended:** Eclipse IDE **Programming language:** JAVA **Hardware requirements:** I3 and I5 machines
REFERENCES	Dhamdhere D., "Systems Programming and Operating Systems", McGraw Hill, ISBN 0 - 07 - 463579 – 4. Find eBook in the following link: https://drive.google.com/drive/folders/1vbu_x7DPHPaIKAbIngKquh5zgYtOgpTe?usp=sharing
STEPS	**Step1. Input for Pass-II of assembler:** 1. IC (in a file) 2. SYMTAB 3. LITTAB 4. POOLTAB **Step2. Open the input IC file and apply the Pass2 of assembler algorithm, process the input file line by line and inside a line token by token** Please refer algorithms of Pass-II of assembler from the following textbook: Dhamdhere D., "Systems Programming and Operating Systems", McGraw Hill, ISBN 0 - 07 - 463579 – 4. Please refer the following link for e-Book and solved test cases **https://drive.google.com/drive/folders/1vbu_x7DPHPaIKAbIngKquh5zgYtOgpTe?usp=sharing** **Step3. Output of Pass-II of assembler:**

	Machine code
INSTRUCTIONS FOR WRITING JOURNAL	1. Date 2. Assignment no. 3. Problem definition 4. Learning objective 5. Learning Outcome 6. Concepts related Theory 7. Algorithm 8. Test cases 9. Conclusion/Analysis

Prerequisites:

Microprocessor, Data structures and Algorithm, Programming and problem solving

Concepts related Theory:

1. Assembler is a low-level translator who translates source code to machine code.
2. It works in two phases: analysis phase and synthesis phase.
3. In analysis phase, source assembly code is analyzed to generate some intermediate data structures.
4. In synthesis phase, machine code is generated.
5. There are well defined system level standard algorithms to design the assembler as a two-pass assembly, namely Pass-I and Pass-II algorithms of assembler.
6. Pass-II takes the intermediate data structures generated by the Pass-I as input and generates machine code.

Algorithm:

Please refer algorithms of Pass-II of assembler from the following textbook: Dhamdhere D., "Systems Programming and Operating Systems", McGraw Hill, ISBN 0 - 07 - 463579 – 4.

Please refer the following link for algorithm in e-Book:

https://drive.google.com/drive/folders/1vbu_x7DPHPaIKAbIngKquh5zgYtOgpTe?usp=sharing

Conclusion:

Generated machine code from intermediate data structures like IC, SYMTAB, LITTAB, POOLTAB.

Review Questions:

1. What is the format of a machine code generated in Pass-II?
2. What is forward reference? How is it resolved by assembler?
3. How is error handling done in pass II?

4. What is the difference between IS, DL and AD?
5. What are the tasks done in Pass II?

ASSINGMENT NO.	A2 (Part 1)
TITLE	Implement pass-I of two-pass macro processor.
PROBLEM STATEMENT/ DEFINITION	Design suitable data structures and implement Pass-I and Pass-II of a two-pass macro- processor. The output of Pass-I (MNT, MDT and intermediate code file without any macro definitions) should be input for Pass-II.
OBJECTIVE	1. Understand the internals of system software viz. i.e., macro processor. 2. Identify and create data structures required in the design of macro processor. 3. Learn parameter processing in macro processor.
OUTCOME	After implementing this assignment, learners will be able to 1. Identify macro definitions and create the intermediate data structures like MNT, MDT, PNTAB, KPDTAB. 2. Understand the relevance of the intermediate data structures with respect to Pass-II of macro processor. 3. Separate macro definitions from the source code.
S/W PACKAGES AND HARDWARE/ APPARATUS USED	**Operating System recommended:** 64-bit Open-source Linux or its derivative **Programming tools recommended:** Eclipse IDE **Programming language:** JAVA **Hardware requirements:** i3/ i5 machines
REFERENCES	Dhamdhere D., "Systems Programming and Operating Systems", McGraw Hill, ISBN 0 - 07 - 463579 – 4. Find eBook in the following link: https://drive.google.com/drive/folders/1vbu_x7DPHPaIKAbIngKquh5zgYtOgpTe?usp=sharing
STEPS	**Step1. Input for Pass-I of macro processor:** 1. Assembly language program in hypothetical language containing macro definitions and macro calls. **Step2. Open the input file and apply the Pass-1 of macro processor algorithm, process the input file line by line and inside a line token by token** Please refer algorithm of Pass-I of macro processor from the following textbook: Dhamdhere D., "Systems Programming and Operating Systems", McGraw Hill, ISBN 0 - 07 - 463579 – 4. Please refer the following link for e-Book and solved test cases **https://drive.google.com/drive/folders/1vbu_x7DPHPaIKAbIngKquh5zgYtOgpTe?usp=sharing** **Step3. Output of Pass-I of macro processor:** 1. MNT 2. MDT 3. PNTAB 4. KPDTAB

INSTRUCTIONS FOR WRITING JOURNAL	1. Assignment no. 2. Title 3. Date of completion 4. Problem definition 5. Learning objectives 6. Learning outcomes 7. Concepts related theory 8. Algorithm 9. Test cases 10. Conclusion/ Analysis 11. Implementation output screenshot

Prerequisites:

Microprocessor, Data structures and Algorithm, Programming and problem solving

Concepts related Theory:

Macro processing feature allows the programmer to write shorthand version of a program (modular programming). The macro processor replaces each macro invocation with the corresponding sequence of statements i.e., macro expansion.

Tasks done by the macro processor

- Recognize macro definitions
- Save the macro definition, recognize macro calls
- Expand macro calls

Tasks in pass-I of a two pass macro processor

- Recognize macro definitions
- Save the macro definition (Create MDT, MNT, PNTAB, KPDTAB).
- Remove macro definitions from the source program.

Algorithm:

1. SSNTAB_ptr:=1; PNTAB_ptr:=1;
2. Process the macro prototype statement and form the MNT entry.
    a) Name:=macro name;
    b) For each positional parameter
        i. Enter parameter name in PNTAB[PNTAB_ptr].
        ii. PNTAB_ptr:=PNTAB_ptr + 1;
        iii. #PP:=#PP+1;
    c) KPDTP:=KPDTAB_ptr;
    d) For each keyword parameter
        i. Enter parameter name and default value (if any) in KPDTAB[KPDTAB_ptr].
        ii. Enter parameter name in PNTAB[PNTAB_ptr].
        iii. KPDTAB_ptr:=KPDTAB_ptr+1;
        iv. PNTAB_ptr:=PNTAB_ptr+1;
        v. #KP:=#KP+1;
    e) MDTP:=MDT_ptr;
    f) #EV:=0;
    g) SSTP:=SSTAB_ptr;

3. While not a MEND statement
    a) If an LCL statement then
        i. Enter expansion time variable name in EVNTAB.
        ii. #EV:=#EV + 1;
    b) If a model statement then
        i. If label field contains a sequencing symbol then
            If symbol is present in SSNTAB then
                q:=entry number in SSNTAB;
        else
            Enter symbol in SSNTAB[SSNTAB_ptr].
            q:=SSNTAB_ptr;
            SSNTAB_ptr:=SSNTAB_ptr + 1;
            SSTAB[SSTP + q -1] := MDT_ptr;
        ii. For a parameter, generate the specification (P,#n)
        iii. For an expansion variable, generate the specification (E,#m).
        iv. Record the IC in MDT[MDT_ptr];
        v. MDT_ptr:=MDT_ptr + 1;

c) If a preprocessor statement then
   i. If a SET statement
      Search each expansion time variable name used in the statement in EVNTAB and
      generate the spec (E,#m).
   ii. If an AIF or AGO statement then
      If sequencing symbol used in the statement is present in SSNTAB Then
         q:=entry number in SSNTAB;
      else
         Enter symbol in SSNTAB[SSNTAB_ptr].
         q:=SSNTAB_ptr;
         SSNTAB_ptr:=SSNTAB_ptr+1;
      Replace the symbol by (S,SSTP + q -1).
   iii. Record the IC in MDT[MDT_ptr]
   iv. MDT_ptr:=MDT_ptr+1;

4. (MEND statement)
   If SSNTAB_ptr=1 (i.e. SSNTAB is empty) then
      SSTP:=0;
   Else
      SSTAB_ptr:=SSTAB_ptr+SSNTAB_ptr-1;
   If #KP=0 then KPDTP=0;

Conclusion:

Input assembly language program containing macros is processed by applying Pass-I algorithm of macro processor and intermediate data structures, MNT, PNTAB, KPDTAB, MDT, are generated.

Review Questions:

1. What is macro and macro processor?
2. What is MDT, MNT?
3. What is nested macro?
4. What are the tasks done in pass I of macro processor?
5. How macro call definitions are handled in pass I?
6. How formal and actual parameters are linked?
7. What are the steps to implement pass I of macro processor?

ASSINGMENT NO.	A2 (Part 2)
TITLE	Implement pass-II of two-pass macro processor.
PROBLEM STATEMENT/ DEFINITION	Design suitable data structures and implement Pass-I and Pass-II of a two-pass macro- processor. The output of Pass-I (MNT, MDT and intermediate code file without any macro definitions) should be input for Pass-II.
OBJECTIVE	1. Understand the internals of system software viz. i.e., macro processor. 2. Understand the macro expansion in Pass-II. 3. Replace the formal parameters with actual parameters. 4. Understand the use of data structures developed in Pass-I.
OUTCOME	After implementing this assignment, learners will be able to 1. Separate the macro definitions from the source code. 2. Generate intermediate data structures like MNT, MDT, PNTAB, KPDTAB. 3. Understand the relevance of the intermediate data structures with respect to Pass-II of macro processor.
S/W PACKAGES AND HARDWARE/ APPARATUS USED	**Operating System recommended:** 64-bit Open-source Linux or its derivative **Programming tools recommended:** Eclipse IDE **Programming language:** JAVA **Hardware requirements:** i3/ i5 machines
REFERENCES	Dhamdhere D., "Systems Programming and Operating Systems", McGraw Hill, ISBN 0 - 07 - 463579 – 4. Find eBook in the following link: https://drive.google.com/drive/folders/1vbu_x7DPHPaIKAbIngKquh5zgYtOgpTe?usp=sharing
STEPS	**Step1. Input for Pass-I of macro processor:** 1. Assembly language program in hypothetical language containing macro definitions and macro calls. **Step2. Open the input file and apply the Pass-1 of macro processor algorithm, process the input file line by line and inside a line token by token** Please refer algorithm of Pass-I of macro processor from the following textbook: Dhamdhere D., "Systems Programming and Operating Systems", McGraw Hill, ISBN 0 - 07 - 463579 – 4. Please refer the following link for e-Book and solved test cases **https://drive.google.com/drive/folders/1vbu_x7DPHPaIKAbIngKquh5zgYtOgpTe?usp=sharing** **Step3. Output of Pass-I of macro processor:** 1. MNT 2. MDT 3. PNTAB 4. KPDTAB

INSTRUCTIONS FOR WRITING JOURNAL	1. Assignment no. 2. Title 3. Date of completion 4. Problem definition 5. Learning objectives 6. Learning outcomes 7. Concepts related theory 8. Algorithm 9. Test cases 10. Conclusion/ Analysis 11. Implementation output screenshot

Prerequisites:

Microprocessor, Data structures and Algorithm, Programming and problem solving

Concepts related Theory:

Macro processing feature allows the programmer to write shorthand version of a program (modular programming). The macro processor replaces each macro invocation with the corresponding sequence of statements i.e., macro expansion.

Tasks done by the macro processor

- Recognize macro definitions
- Save the macro definition, recognize macro calls
- Expand macro calls

Tasks in pass-I of a two pass macro processor

- Recognize macro definitions
- Save the macro definition (Create MDT, MNT, PNTAB, KPDTAB).
- Remove macro definitions from the source program.

Algorithm:

1. SSNTAB_ptr:=1; PNTAB_ptr:=1;
2. Process the macro prototype statement and form the MNT entry.
    a) Name:=macro name;
    b) For each positional parameter
        i. Enter parameter name in PNTAB[PNTAB_ptr].
        ii. PNTAB_ptr:=PNTAB_ptr + 1;
        iii. #PP:=#PP+1;
    c) KPDTP:=KPDTAB_ptr;
    d) For each keyword parameter
        i. Enter parameter name and default value (if any) in KPDTAB[KPDTAB_ptr].
        ii. Enter parameter name in PNTAB[PNTAB_ptr].
        iii. KPDTAB_ptr:=KPDTAB_ptr+1;
        iv. PNTAB_ptr:=PNTAB_ptr+1;
        v. #KP:=#KP+1;
    e) MDTP:=MDT_ptr;
    f) #EV:=0;
    g) SSTP:=SSTAB_ptr;

3. While not a MEND statement
    a) If an LCL statement then
        i. Enter expansion time variable name in EVNTAB.
        ii. #EV:=#EV + 1;
    b) If a model statement then
        i. If label field contains a sequencing symbol then
            If symbol is present in SSNTAB then
                q:=entry number in SSNTAB;
            else
                Enter symbol in SSNTAB[SSNTAB_ptr].
                q:=SSNTAB_ptr;
                SSNTAB_ptr:=SSNTAB_ptr + 1;
                SSTAB[SSTP + q -1] := MDT_ptr;
        ii. For a parameter, generate the specification (P,#n)
        iii. For an expansion variable, generate the specification (E,#m).
        iv. Record the IC in MDT[MDT_ptr];
        v. MDT_ptr:=MDT_ptr + 1;

c) If a preprocessor statement then
   i. If a SET statement
      Search each expansion time variable name used in the statement in EVNTAB and
      generate the spec (E,#m).
   ii. If an AIF or AGO statement then
      If sequencing symbol used in the statement is present in SSNTAB Then
         q:=entry number in SSNTAB;
      else
         Enter symbol in SSNTAB[SSNTAB_ptr].
         q:=SSNTAB_ptr;
         SSNTAB_ptr:=SSNTAB_ptr+1;
      Replace the symbol by (S,SSTP + q -1).
   iii. Record the IC in MDT[MDT_ptr]
   iv. MDT_ptr:=MDT_ptr+1;

4. (MEND statement)
   If SSNTAB_ptr=1 (i.e. SSNTAB is empty) then
      SSTP:=0;
   Else
      SSTAB_ptr:=SSTAB_ptr+SSNTAB_ptr-1;
   If #KP=0 then KPDTP=0;

Conclusion:

All the macro calls in the assembly program are processed using the output of Pass-I algorithm of macro processor and intermediate data structures, MNT, PNTAB, KPDTAB, MDT. APTAB generated in the Pass-II of macro processor.

Review Questions:

1. What is macro and macro processor?
2. What is MDT, MNT?
3. What is nested macro?
4. What are the tasks done in pass I of macro processor?
5. How macro call definitions are handled in pass I?
6. How formal and actual parameters are linked?
7. What are the steps to implement pass I of macro processor?

ASSINGMENT NO.	B1
TITLE	Synchronization Using Mutex and Semaphore
PROBLEM STATEMENT /DEFINITION	Write a program to solve Classical Problems of Synchronization using Mutex and Semaphore
OBJECTIVE	To learn and understand 1. To Implement Process Synchronization using Mutex & Semaphore 2. To Solve Classical Synchronization problems using Mutex & Semaphore
OUTCOME	Refer to details
S/W PACKAGES AND HARDWARE Requirement	C/C++ editors and compilers for Linux OS Linux OS/Fedora/Ubuntu PC i3/i5/i7 2.4 GHz. 4/8 GB RAM, 500/1TB G.B HDD,Mouse and Keyboard
REFERENCES	**1.** Stallings W., "Operating Systems", 4th Edition, Prentice Hall, 81 – 7808 – 503 – 8. **2.** Silberschatz, Galvin, Gagne, "Operating System Principles", 9th Edition, Wiley, ISBN 978-1-118-06333-0.
STEPS	Refers To Details
INSTRUCTIONS FOR WRITING JOURNAL	1. Date 2. Assignment no. 3. Problem definition 4. Learning objective 5. Learning Outcome 6. Concepts related Theory 7. Algorithm 8. Test cases 9. Conclusion/Analysis

Prerequisites: Semaphore concepts, Producer consumer problem, Reader Writer problem

Concepts related Theory:

A process can stop at a specified place until it has received a specific signal. For signaling, special variables called Semaphores are used. The wait() operation decrements the semaphore value. If the

value becomes negative, then the process executing the wait() is blocked. The signal() operation increments the semaphore value. If the value is not positive, then a process is unblocked.

```
struct semaphore {
int count;
queuetype queue;
}
void wait (semaphore s) {
s.count --; //decrement
if (s.count < 0) {
place this process in s.queue;
block this process;
}
}

void signal (semaphore s) {
s.count ++; //increment
if (s.count < = 0) {
remove a process P from s.queue;
place P on ready queue;
}
}
```

Mutex is an abbreviation for "mutual exclusion". Mutex variables are one of the primary means of implementing thread synchronization and for protecting shared data when multiple writes occur. A mutex variable acts like a "lock" protecting access to a shared data resource. The basic concept of a mutex as used in Pthreads is that only one thread can lock (or own) a mutex variable at any given time. Thus, even if several threads try to lock a mutex only one thread will be successful. No other thread can own that mutex until the owning thread unlocks that mutex.

wait (mutex); *//locks*

…..

Critical Section

…..

signal (mutex); *//unlocks*

The producer-consumer problem (Also called the bounded-buffer problem.) illustrates the need for synchronization in systems where many processes share a resource. In the problem, two processes share a fixed-size buffer. One process(producer) produces information and puts it in the buffer, while the other process (consumer) consumes information from the buffer. These processes do not take turns accessing the buffer, they both work concurrently. Herein lies the problem. What happens if the producer tries to put an item into a full buffer? What happens if the consumer tries to take an item from an empty buffer?

In order to synchronize these processes, we will block the producer when the buffer is full, and we will

block the consumer when the buffer is empty. So, the two processes, Producer and Consumer, should work as follows:

Algorithm:

Assuming there are total N number of slots.

Initialization: semaphores: mutex = 1; Full = 0 , empty = N;integers : in = 0; out = 0;

producer:

```
Repeat forever
produce (item);
wait(empty);
wait(mutex);
enter_item(item)
; signal(mutex);
signal(full);
```

Consumer:

```
Repeat forever
wait(full);
wait(mutex);
remove_item(item);
signal( mutex );
signal(empty);
```

POSIX: POSIX stands for Portable Operating System Interface

sem_init()
Initializes a semaphore

```
#include
<semaphore.h>
int sem_init(sem_t *sem, int pshared, unsigned int value);
```

- First argument is the pointer to semaphore, that you want to initialize. sem_init initializes the semaphore object pointed to by sem.
- The pshared argument indicates whether the semaphore is local to the current process (pshared is zero) or is to be shared between several processes (pshared is not zero). LinuxThreads currently does not support process-shared semaphores, thus sem_init alwaysreturns with error ENOSYS if pshared is not zero.
- Third argument is the value of the semaphore. The count associated with the semaphore is set initially to value.

pthread_create()

```
#include <pthread.h>
int pthread_create(pthread_t * thread, pthread_attr_t * attr, void *(*start_routine)(void *), void
* arg);
```

pthread_create creates a new thread of control that executes concurrently with the calling thread.

The new thread applies the function start_routine passing it arg as first argument.

The new thread terminates either explicitly, by calling pthread_exit(3), or implicitly, by returning from the start_routine function. The latter case is equivalent to calling pthread_exit(3) with the result returned by start_routine as exit code.

The attr argument specifies thread attributes to be applied to the new thread.

pthread_join()

```
#include <pthread.h>
int pthread_join(pthread_t th, void **thread_return);
```

pthread_join suspends the execution of the calling thread until the thread identified by **th** terminates, either by calling pthread_exit(3) or by being cancelled. If thread_return is not NULL, the return value of th is stored in the location pointed to by thread_return. The return value of **th** is either the argument it gave to pthread_exit(3), or PTHREAD_CANCELED if th was cancelled.

The joined thread th must be in the joinable state.

Conclusion: Producer consumer problem is solved successfully using POSIX threads and semaphore.

Review Questions:

1. What is race condition?
2. What is multi-threading? write advantages of multi-threading
3. What is Critical Section?
4. What is process Synchronization?
5. Define Mutex And Semaphore.
6. Solve Reader writer problem using Mutex and Semaphore

ASSINGMENT NO.	B2
TITLE	Implement CPU Scheduling Algorithms.
PROBLEM STATEMENT/ DEFINITION	Write a program to simulate CPU Scheduling Algorithms: FCFS, SJF (Preemptive), Priority (Non-Preemptive) and Round Robin (Preemptive).
OBJECTIVE	To learn and understand 1. Process Scheduling in Multitasking and multiusers OS. 2. Implementation of Scheduling Algorithms.
OUTCOME	After implementing this assignment, learners will be able to 1. Implement FCFS, SJF, RR, Priority Scheduling Algorithms 2. Analyze and compare performance of scheduling algorithms.
S/W PACKAGES AND HARDWARE/ APPARATUS USED	**Operating System recommended:** 64-bit Open-source Linux or its derivative **Programming tools recommended:** Eclipse IDE **Programming language:** C/ C++/ JAVA/ Python **Hardware requirements:** i3/ i5 machines
REFERENCES	Stallings W., "Operating Systems", 4th Edition, Prentice Hall, 81 – 7808 – 503 – 8.
STEPS	Refer to details.
INSTRUCTIONS FOR WRITING JOURNAL	1. Assignment no. 2. Title 3. Date of completion 4. Problem definition 5. Learning objectives 6. Learning outcomes 7. Concepts related theory 8. Algorithm 9. Test cases 10. Conclusion/ Analysis 11. Implementation output screenshot

Prerequisites:

Basic Operating System Functionalities.

Concept of Multitasking and Multiuser OS.

THEORY:

Process scheduling: It is an activity process manager that handles the task of scanning process from

CPU and running of another process on basis of some strategy. Such OS allows more than one to be loaded in executable memory.

Scheduler:
They are special system software that handle process scheduling in various ways:
Its main task is to select jobs to be submitted into system and to be divided and decide which process to run.

Types of Schedulers:

1. Long-term scheduler
2. Short-term scheduler
3. Medium term scheduler

Arrival Time: The request arrives in the system when user submits it to OS. When request arrives, the time is called arrival time.
Scheduling: The pending request is scheduled for service when scheduler selects it for servicing.
Pre-emption: The process is pre-empted when CPU switches to another process before completing it and this process is added to pending request.
Non pre-emption: The scheduled process is always completed before next scheduling of the process.
Completion: The process is completed, and next process is selected for processing by CPU.
The CPU scheduling takes the information about arrival time, size of request in CPU seconds, CPU time already consumed by the request, deadline of the process for scheduling policy.
CPU scheduling deals with the problem of deciding which of the processes in the ready queue is to be allowed to utilize the CPU. The criteria for selection of an algorithm are

- Least turnaround time.
- Minimum waiting time.
- Maximum CPU utilization.
- The maximum throughput

Some definitions in scheduling are:
Arrival time is when the process arrives in the system. (Ai)
Process time is the execution time required for the process (Xi)
Completion time is time at which the process is completed. (Ci)
Deadline is the time by which the process output is required (Di)
Turnaround time is the time to complete the process after arrival (Ci-Ai)
Average or Mean turnaround time is the average of turnaround time of all processes.
Weighted time around time is the turnaround time of a process to its execution time. (Ci-Ai)/Xi
Throughput is the measure of performance and no of processes completed per unit time. (n/(max (Ci)-min(Ai))

Scheduling Policies:

FCFS Scheduling:

The process requests are scheduled in the order of their arrival time. The pending requests are in a queue. The first request in the queue is scheduled first. The request that comes is added to the end of

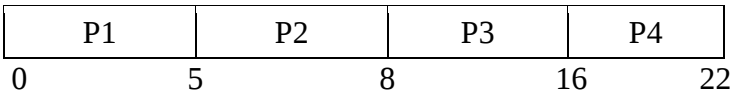
the queue.

Performance of FCFS scheduling: (Time in sec)

Process No	Arrival time	Burst Time	Turnaround time	Waiting time
1	0	5	0	0
2	1	3	5	4
3	2	8	8	6
4	3	6	16	13

Average waiting time= (0+4+6+13)/4= 5.75

Gantt Chart:



Algorithm:

1. Input the processes along with burst times.
2. Input arrival time for all processes
3. Sort according to their arrival time along with indices.
4. Perform processes in sorted order
5. Stop.

Shortest job first (SJF) scheduling:

Best approach to minimize waiting time. It is easy to implement in batch systems where required CPU time is known in advance.

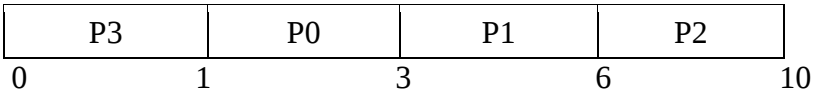
a) **Non pre-emptive:**

Performance of SJF scheduling: (Time in sec)

Process No	Arrival time	Burst time	Turnaround time	Waiting time
1	0	2		1
2	0	3		3
3	1	4		5
4	0	1		0

Average waiting time= 2.25

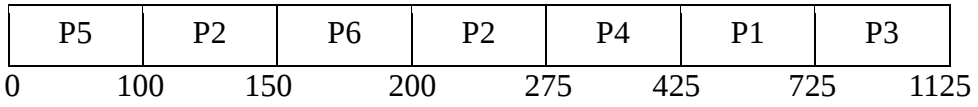
Gantt Chart:



b) **Pre-emptive:**

Process	Arrival time	Burst time	Waiting time
P1	0	300	425
P2	0	125	150
P3	0	400	725
P4	0	150	275
P5	0	100	0
P6	150	50	0

Gantt Chart:



Average waiting time= 262.5

Algorithm

1. Calculate burst time
2. Sort all processes in increasing order of burst time
3. Apply FCFS to sorted list
4. Perform all processes
5. Stop

Round Robin scheduling:

Schedules using time slicing. The amount of CPU time a process may use when allocated is limited. The process is pre-empted if the process requires more time or if process requires I/O operation before the time slice. It makes weighted turnaround time approximately equal all time, but throughput may not be well as all processes are treated equally.

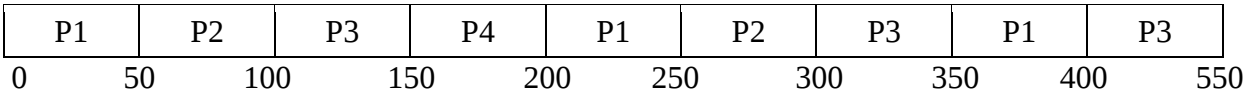
Performance of Round Robin scheduling:

Process No	Arrival time Ai	Burst time	Waiting time
1	1	150	250
2	2	100	200
3	3	200	300
4	4	50	150

Time quantum= 50

Average waiting time= 225

Gantt Chart:



Algorithm:

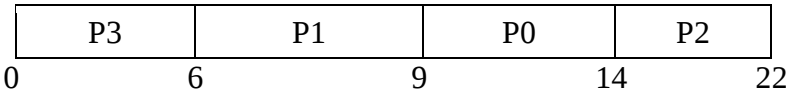
1. Get input for processes with arrival time and burst time. Take time quantum.
2. Sort processes according to arrival time.
3. Process till all processes are done.
4. End

Priority based Scheduling:

It is non-preemptive algorithm and one of the common scheduling algorithm in batch system. Each process is assigned a priority and process with highest priority is executed first and so on. Processes with same priority are executed on FCFS basis.

Process	Arrival time	Burst time	Priority	Waiting time
P0	0	5	1	9
P1	1	3	2	5
P2	2	8	1	12
P3	3	6	3	0

Gantt Chart:



Algorithm:

1. Get input for process including arrival time, burst time and priority.
2. Sort process according to arrival time.
3. If process have same arrival time, sort them by priority.
4. Print process according to index.
5. End

Steps to do /algorithm:

1. Create a menu to select various scheduling algorithms
2. Take number of tasks and CPU time as input.
3. Calculate average waiting time and turnaround time for each scheduling strategy.
4. Perform a comparative assessment of best policy for given set of processes.

FAQs

1. What are the inputs to be taken?

Ans: The inputs for each process with process no, Arrival time, Execution time to be taken.

2. What all parameters to be calculated?

Ans: The turnaround time, Waiting time, Average Turnaround Time, Average Waiting Time.

3. How the output to be shown?

Ans: The output for each algorithm for the same set of inputs to be shown as table and Gant chart to be shown in write-up.

Oral/ Review Questions:

1. Why is job scheduling required in OS?
2. What are basic job scheduling policies used in OS?
3. What is preemptive scheduling?
4. What is the computational complexity of RR, SJF, FCFS?
5. What is Gant chart?

ASSINGMENT NO.	B4
TITLE	Simulate Page replacement algorithm.
PROBLEM STATEMENT/ DEFINITION	Write a program to simulate Page replacement algorithm. 1. FIFO 2. Least Recently Used (LRU) 3. Optimal
OBJECTIVE	1. To understand virtual memory management. 2. To analyze the need of page replacement algorithms. 3. To compare various page replacement algorithms.
OUTCOME	After implementing this assignment, learners will be able to 1. Implement page replacement algorithms viz. FIFO, LRU and Optimal. 2. Compare the performance of page replacement algorithms based on hit ratio.
S/W PACKAGES AND HARDWARE/ APPARATUS USED	**Operating System recommended:** 64-bit Open-source Linux or its derivative **Programming tools recommended:** Eclipse IDE **Programming language:** C/ C++/ JAVA/ Python **Hardware requirements:** i3/ i5 machines
REFERENCES	Stallings W., "Operating Systems", 4th Edition, Prentice Hall, 81 – 7808 – 503 – 8.
STEPS	Refer to details.
INSTRUCTIONS FOR WRITING JOURNAL	1. Assignment no. 2. Title 3. Date of completion 4. Problem definition 5. Learning objectives 6. Learning outcomes 7. Concepts related theory 8. Algorithm 9. Test cases 10. Conclusion/ Analysis 11. Implementation output screenshot

Theory:

What is Page Replacement in Operating Systems?

Page replacement is needed in the operating systems that use virtual memory using Demand Paging. As we know that in Demand paging, only a set of pages of a process is loaded into the memory. This is done so that we can have more processes in the memory at the same time.

When a page that is residing in virtual memory is requested by a process for its execution, the Operating System needs to decide which page will be replaced by this requested page. This process is

known as page replacement and is a vital component in virtual memory management.

Why we need page replacement algorithms?

To understand why we need page replacement algorithms, we first need to know about page faults.

Page Fault: A Page Fault occurs when a program running in CPU tries to access a page that is in the address space of that program, but the requested page is currently not loaded into the main physical memory, the RAM of the system.

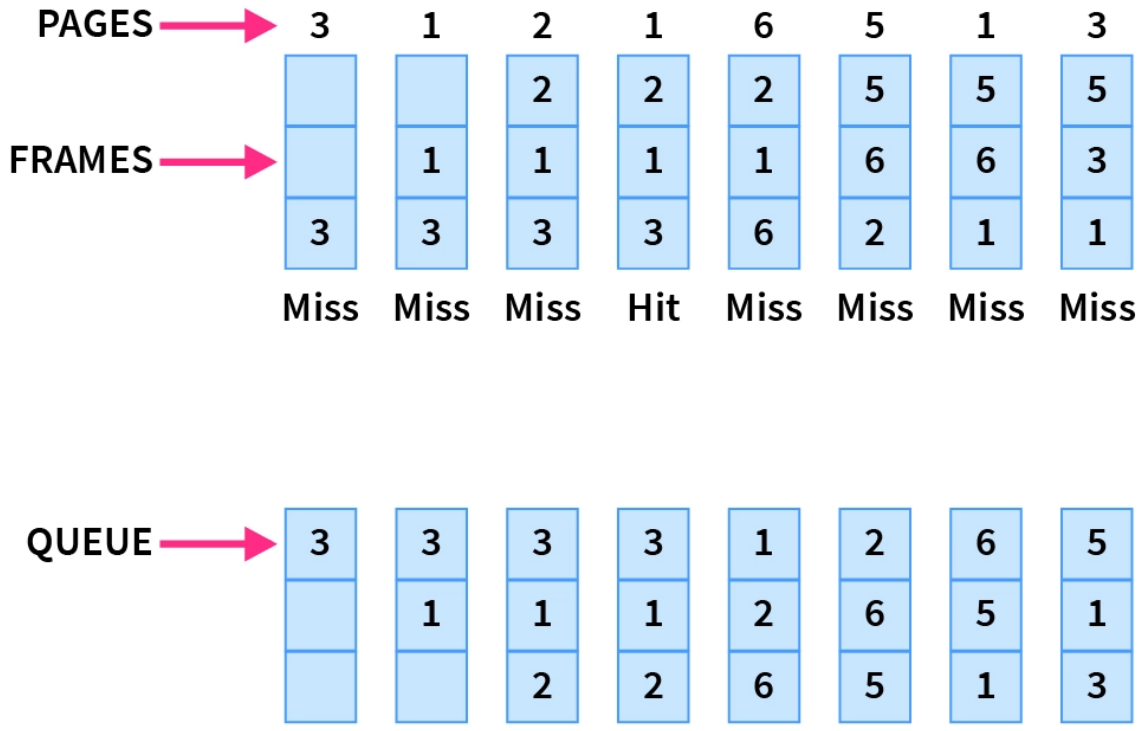
Since the actual RAM is much less than the virtual memory the page faults occur. So, whenever a page fault occurs, the Operating system has to replace an existing page in RAM with the newly requested page. In this scenario, page replacement algorithms help the Operating System in deciding which page to replace. The primary objective of all the page replacement algorithms is to minimize the number of page faults.

First In First Out (FIFO) Page Replacement Algorithm

FIFO algorithm is the simplest of all the page replacement algorithms. In this, we maintain a queue of all the pages that are in the memory currently. The oldest page in the memory is at the front-end of the queue and the most recent page is at the back or rear-end of the queue.

Whenever a page fault occurs, the operating system looks at the front-end of the queue to know the page to be replaced by the newly requested page. It also adds this newly requested page at the rear-end and removes the oldest page from the front-end of the queue.

Example: Consider the page reference string as 3, 1, 2, 1, 6, 5, 1, 3 with 3-page frames. Let's try to find the number of page faults:



Total page faults = 7

Belady's anomaly:

Generally, if we increase the number of frames in the memory, the number of page faults should decrease due to obvious reasons. Belady's anomaly refers to the phenomena where increasing the number of frames in memory, increases the page faults as well.

Advantages of FIFO policy:
Simple to understand and implement
Does not cause more overhead

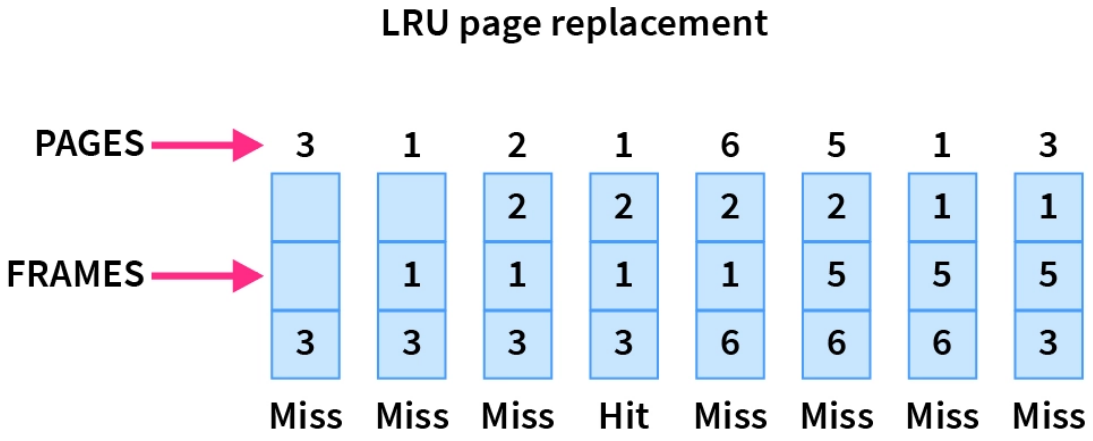
Disadvantages of FIFO policy:
Poor performance
Doesn't use the frequency of the last used time and just simply replaces the oldest page.
Suffers from Belady's anomaly.

Least Recently Used (LRU) Page Replacement Algorithm

The least recently used page replacement algorithm keeps the track of usage of pages over a period of time. This algorithm works on the basis of the principle of locality of a reference which states that a program has a tendency to access the same set of memory locations repetitively over a short period of time. So, pages that have been used heavily in the past are most likely to be used heavily in the future also.

In this algorithm, when a page fault occurs, then the page that has not been used for the longest duration of time is replaced by the newly requested page.

Example:
Let's see the performance of the LRU on the same reference string of 3, 1, 2, 1, 6, 5, 1, 3 with 3-page frames:



Total page faults = 7

Now in the above example, the LRU causes the same page faults as the FIFO, but this may not always be the case as it will depend upon the series, the number of frames available in memory, etc. In fact, on most occasions, LRU is better than FIFO.

Advantages of LRU policy:
It is open for full analysis
Doesn't suffer from Belady's anomaly
Often more efficient than other algorithms

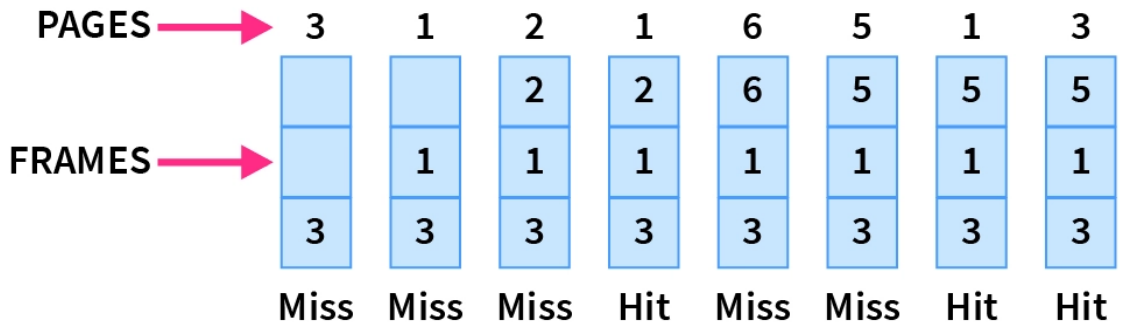
Disadvantages of LRU policy:
It requires additional data structures to be implemented

More complex
High hardware assistance is required

Optimal Page Replacement Algorithm:

Optimal page replacement is the best page replacement algorithm as this algorithm results in the least number of page faults. In this algorithm, the pages are replaced with the ones that will not be used for the longest duration of time in the future. In simple terms, the pages that will be referred farthest in the future are replaced in this algorithm.

Example:
Let's take the same page reference string 3, 1, 2, 1, 6, 5, 1, 3 with 3-page frames as we saw in FIFO. This also helps you understand how Optimal Page replacement works the best.



Total page faults = 5

Optimal page replacement algorithm is difficult to implement, because it requires future knowledge of the reference string. As a result, the optimal algorithm is used mainly for comparison studies. For instance, it may be useful to know that, although a new algorithm is not optimal, it is within 12.3 percent of optimal at worst, and within 4.7 percent on average.

Advantages of Optimal policy:
Excellent efficiency
Less complexity
Easy to use and understand
Simple data structures can be used to implement
Used as the benchmark for other algorithms

Disadvantages of optimal policy:
More time consuming
Difficult for error handling
Need future awareness of the programs, which is not possible every time

Conclusion:
The objective of page replacement algorithms is to minimize the page faults.
FIFO page replacement algorithm replaces the oldest page in the memory.
Optimal page replacement algorithm replaces the page which will be referred farthest in the future.
LRU page replacement algorithm replaces the page that has not been used for the longest duration of time.

PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE

ACADEMIC YEAR: 2023-24

DEPARTMENT of COMPUTER ENGINEERING DEPARTMENT

CLASS: T.E. **SEMESTER: I**

SUBJECT: LP-1 (IoT & ES)

ASSIGNMENT NUMBER: 1

Revised On: 18-07-2022

TITLE	Connectivity of Raspberry Pi/Arduino with InfraRed (IR)sensor.
PROBLEM STATEMENT /DEFINITION	Understanding the connectivity of Raspberry-Pi/Arduino with IR sensor. Write an application to detect obstacle and notify user using LEDs.
OBJECTIVE	Understanding the connectivity of Raspberry Pi/ Arduino circuit withIR sensor.
S/W PACKAGES AND HARDWARE APPARATUS USED	Raspberry pi board/Arduino, IR sensor, LED Raspbian (OS),
REFERENCES	1. https://www.raspberrypi.org/
STEPS	Refer to details
INSTRUCTIONS FOR WRITING JOURNAL	• Title • Problem Definition • Objectives • Theory • Interfacing Diagram • Algorithm • Test cases • Program Listing (soft copy) • Output • Conclusion

Aim: Connectivity of Raspberry Pi / Arduino circuit with IR sensor to detect obstacleand notify user using LEDs.

Pre-requisite:

Basic knowledge of GPIO of Raspberry pi/ Arduino.
Basic knowledge of Python programming.
Working and connections of sensors.

Learning Objectives:

- Understanding the connectivity of Raspberry Pi /Arduino with IR sensor.

Learning Outcomes:

The students will be able to

- To interface IR sensor to Raspberry pi.
- Detect the obstacle with IR sensor.
- Can perform actuation.

H/W AND S/W Requirements:

Raspberry Pi/ Arduino Boards

PC / Monitor/Keyboard

IR (Infrared) Sensor, 1 LED, 1 Resistor (330 Ω)

Few jumper cables,1 Breadboard
Raspbian (OS), Debian LINUX and Python

Theory:

Introduction:

The Raspberry Pi is a series of credit card-sized single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote the teaching of basic computer science in schools and in developing countries. It is a capable little computer which can be used in electronics projects, and for many of the things that your desktop PC does, like spreadsheets, word processing, browsing the internet, and playing games. It also plays high-definition video.

The Raspberry Pi is open hardware, with the exception of the primary chip on the Raspberry Pi, the Broadcom SoC (System on a Chip), which runs many of the main components of the board–CPU, graphics, memory, the USB controller, etc. Many of the projects made with a

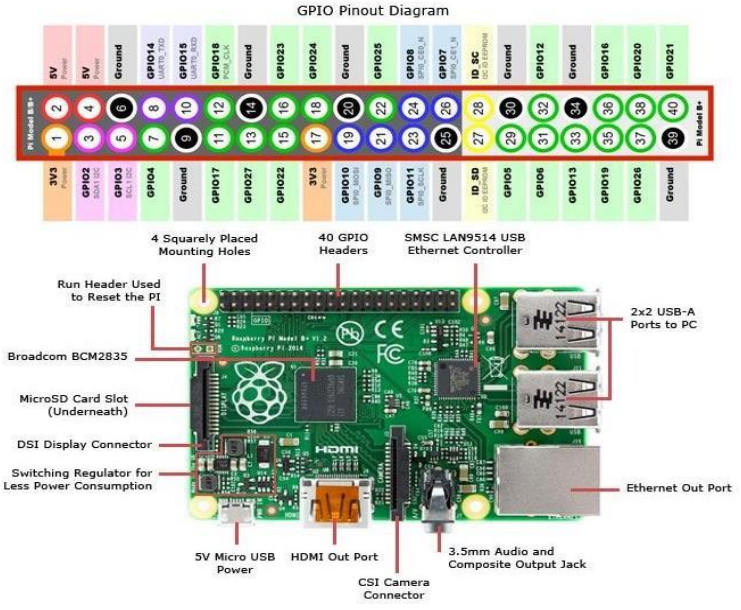
Raspberry Pi are open and well-documented as well and are things you can build and modify yourself.

The Raspberry Pi was designed for the Linux operating system, and many Linuxdistributions now have a version optimized for the Raspberry Pi.

One powerful feature of the Raspberry Pi is the row of GPIO (general purpose input/output) pins along the top edge of the board. These pins are a physical interface between the Pi and the outside world. At the simplest level, you can think of them as switches that you can turn on or off (input) or that the Pi can turn on or off (output). Of the 40 pins, 26 are GPIO pinsand the others are power or ground pins (plus two ID EEPROM pins which you should not play with unless you know your stuff!)

You can program the pins to interact in amazing ways with the real world. Inputs don't have to come from a physical switch; it could be input from a sensor or a signal from another computer or device, for example. The output can also do anything, from turning on an LED to sending a signal or data to another device. If the Raspberry Pi is on a network, you can control devices that are attached to it from anywhere and those devices can send data back. Connectivity and control of physical devices over the internet is a powerful and exciting thing, and the Raspberry Pi is ideal for this.

Raspberry Pi Board with GPIO:

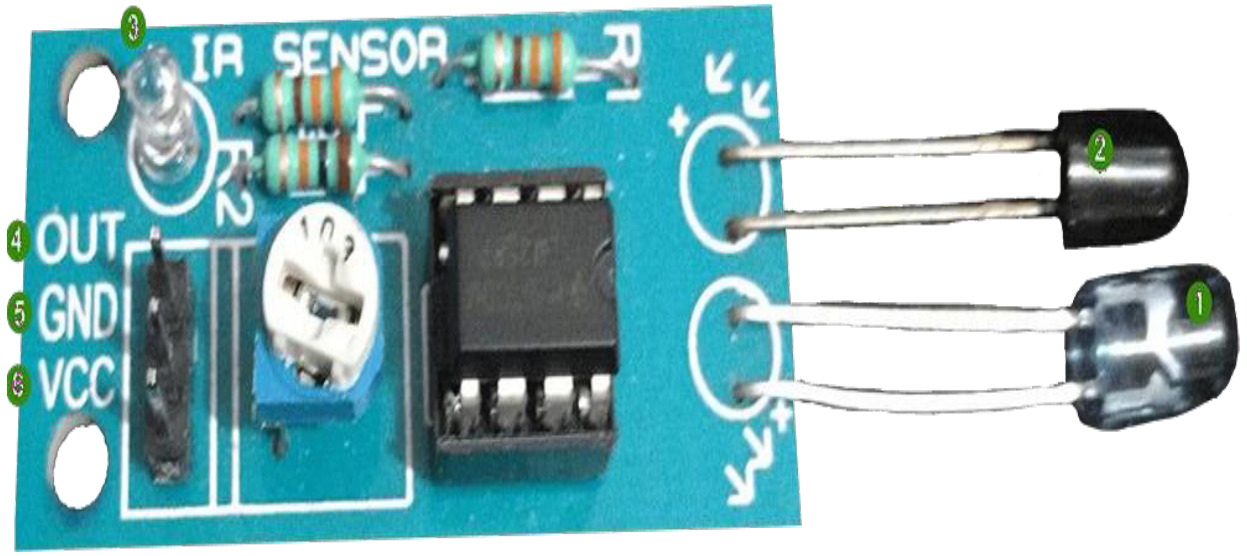


Technical Specification:

- Broadcom BCM2837 64bit ARMv7 Quad Core Processor powered Single Board
- Computer running at 1.2GHz
- 1GB RAM
- BCM43143 WiFi on board
- Bluetooth Low Energy (BLE) on board
- 40pin extended GPIO
- 4 x USB 2 ports
- 4 pole Stereo output and Composite video port
- Full size HDMI
- CSI camera port for connecting the Raspberry Pi camera
- DSI display port for connecting the Raspberry Pi touch screen display
- Micro SD port for loading your operating system and storing data
- Upgraded switched Micro USB power source (now supports up to 2.4 Amps)
- Expected to have the same form factor has the Pi 2 Model B, however the LEDs willChange position

InfraRed (IR) Sensor:

IR (Infrared) Sensor works by emitting infrared signal/radiation and receiving of the signal when the signal bounces back from any obstacle. In other words, the IR Sensor works by continuously sending signal (in a direction) and continuously receive signal, if comes back by bouncing on any obstacle in the way.



Components: IR Sensor

1. **Emitter**: This component continuously emits the infrared signal
2. **Receiver**: It waits for the signal which is bounced back by obstacle
3. **Indicator**: On board LED to signal if obstacle is deducted by the sensor
4. **Output**: Could be used as Input for further processing of the signal
5. **Ground**: Ground/Negative point of the circuit
6. **Voltage**: Input 3.3V

IR Sensor has 3 pins, viz VCC, GND and OUT. We will use GPIO 17 (do not get confused with pin number 17) for receiving input from the sensor.

Connecting IR Sensor

1. Connect GPIO 17 from the Raspberry Pi to Breadboard
2. Connect OUT pin of the sensor with the Breadboard
   This will send input received from sensor to GPIO 17, which will be processed further.
3. Connect GND (any pin from board will work) with negative line on left side of the breadboard
4. Connect GND of the IR Sensor to Breadboard
5. Connect GND from Step 3 to Breadboard
6. Connect VCC of the IR Sensor to Breadboard
7. Connect 3v3 (Pin #1) to positive line on left side of the breadboard
8. Connect 3v3 (connected in Step 7) to the Breadboard

Connecting LED

Objective is to turn on the LED when obstacle is detected.

1. Connect GPIO 4 from the board to the Breadboard
2. Connect positive point of the LED (longer pin of the LED) to the Breadboard
3. Connect negative point of the LED (smaller pin of the LED) to the Breadboard
4. Use resistor (330 Ω) to connect negative to the negative point of the LED

Python code to detect obstacle

```
from gpiozero import LED
from signal import pause

import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

LED_PIN = 27
IR_PIN = 17
indicator = LED(LED_PIN)
GPIO.setup(IR_PIN, GPIO.IN)
```

```
count = 1

while True:
  got_something = GPIO.input(IR_PIN)
  if got_something:
    indicator.on()
    print("{:>3} Got something".format(count))
  else:
    indicator.off()
    print("{:>3} Nothing detected".format(count))
  count += 1
  time.sleep(0.2)
```

Conclusion:

Students will be able to interface IR sensor with Raspberry Pi and detect an obstacle with it.

FAQs:

1. What is IR sensor?
2. How is it interfaced with Raspberry pi/Arduino?

ASSIGNMENT NUMBER: 2

Revised On: 18-07-2022

TITLE	Connectivity of Raspberry Pi /Beagle board circuit with temperaturesensor.
PROBLEM STATEMENT /DEFINITION	Write an application to read the environment temperature. If temperature crosses a threshold value, the application indicated user using LEDSs
OBJECTIVE	Understanding the connectivity of Raspberry Pi /Beagle board circuitwith temperature sensor.
S/W PACKAGES AND HARDWARE APPARATUS USED	Raspberry pi board/ BBB , DTH-11 temperature sensor,LED Raspbian (OS), Adafruit_DTH Library
REFERENCES	https://www.raspberrypi.org/ https://www.bbb.org/ https://www.adafruit.com/
STEPS	Refer to details
INSTRUCTIONSFOR WRITINGJOURNAL	Title Problem Definition Objectives Theory Interfacing Diagram Algorithm Test cases Program Listing (soft copy) Output Conclusion

Aim:

Connectivity of Raspberry Pi /Beagle board circuit with temperature sensor to readthe environment temperature. If temperature crosses a threshold value, the application indicated user using LEDs.

Pre-requisite:

- Basic knowledge of GPIO of Raspberry pi/BBBBasic knowledge of Python programming.
- Working and connections of sensors.

Learning Objectives:

- Understanding the connectivity of Raspberry Pi /Beagle board circuit with temperature sensor.

Learning Outcomes:

The students will be able to

- To interface temperature sensor to Raspberry pi.
- Read and analyze temperature values.
- Can perform actuation.

Theory:

Introduction:

The Raspberry Pi is a series of credit card-sized single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote the teaching of basic computer science in schools and in developing countries. It is a capable little computer which can be used in electronics projects, and for many of the things that your desktop PC does, like spreadsheets, word processing, browsing the internet, and playing games. It also plays high- definition video.

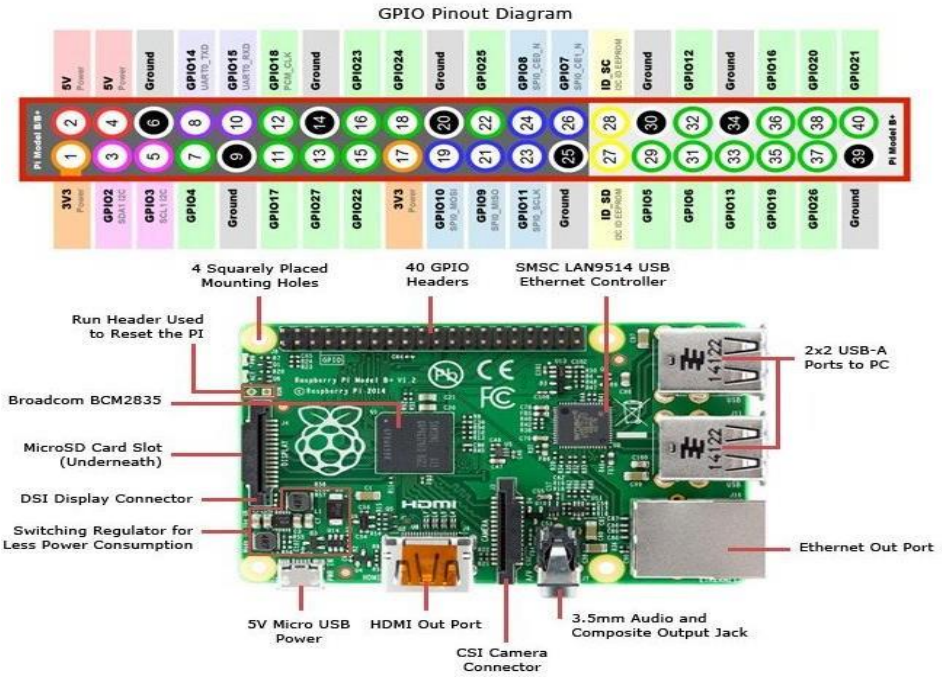
The Raspberry Pi is open hardware, with the exception of the primary chip on the Raspberry Pi, the Broadcom SoC (System on a Chip), which runs many of the main components of the board–CPU, graphics, memory, the USB controller, etc. Many of the projects made with a Raspberry Pi are open and well-documented as well and are things you can build and modify yourself.

The Raspberry Pi was designed for the Linux operating system, and many Linux distributions now have a version optimized for the Raspberry Pi.

One powerful feature of the Raspberry Pi is the row of GPIO (general purpose input/output) pins along the top edge of the board. These pins are a physical interface between the Pi and the outside world. At the simplest level, you can think of them as switches that you can turn on or off (input) or that the Pi can turn on or off (output). Of the 40 pins, 26 are GPIO pinsand the others are power or ground pins (plus two ID EEPROM pins which you should not play with unless you know your stuff!)

You can program the pins to interact in amazing ways with the real world. Inputs don't haveto come from a physical switch; it could be input from a sensor or a signal from another computer or device, for example. The output can also do anything, from turning on an LEDto sending a signal or data to another device. If the Raspberry Pi is on a network, you can control devices that are attached to it from anywhere and those devices can send data back. Connectivity and control of physical devices over the internet is a powerful and exciting thing, and the Raspberry Pi is ideal for this.

Raspberry Pi Board with GPIO:

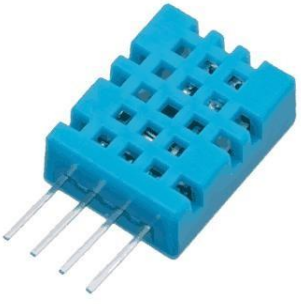


Technical Specification:

Broad com BC M28 37 64bit AR Mv7 Quad Core Processor powered Single Board
Computer running at 1.2GHz
1GB RAM
BCM43143 WiFi on board
Bluetooth Low Energy (BLE) on board
40pin extended GPIO
4 x USB 2 ports
4 pole Stereo output and Composite video port
Full size HDMI
CSI camera port for connecting the Raspberry Pi camera
DSI display port for connecting the Raspberry Pi touch screen display
Micro SD port for loading your operating system and storing data
Upgraded switched Micro USB power source (now supports up to 2.4 Amps)
Expected to have the same form factor has the Pi 2 Model B, however the LEDs willChange position

DHT11 Sensor

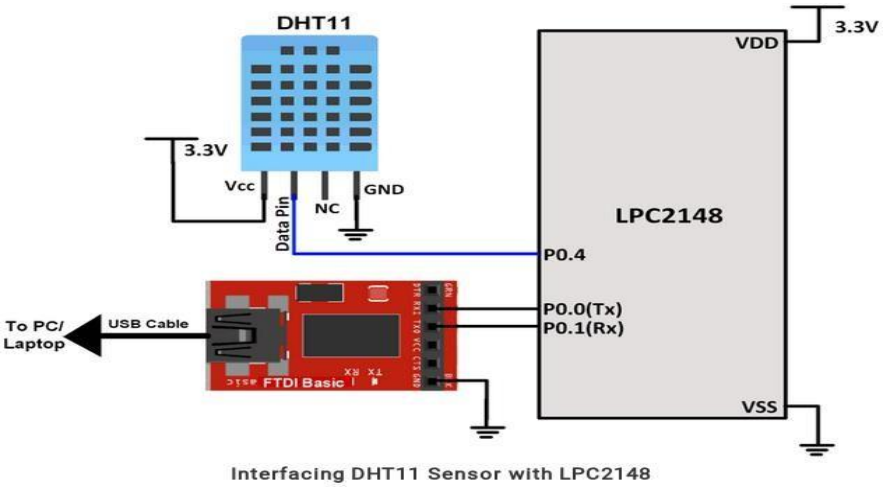
Introduction



DHT11 Sensor

DHT11 is a single wire digital humidity and temperature sensor, which provides humidity and temperature values serially.
It can measure relative humidity in percentage (20 to 90% RH) and temperature in degreeCelsius in the range of 0 to 50°C.
It has 4 pins of which 2 pins are used for supply, 1 is not used and the last one is used fordata.
The data pin is the only pin used for communication. Pulses of different TON and TOFF aredecoded as logic 1 or logic 0 or start pulse or end of the frame.
For more information about DHT11 sensor and how to use it, refer the topic DHT11 sensor inthe sensors and modules topic.

Interfacing Diagram:



Interfacing DHT11 Sensor with LPC2148

Conclusion:
Students will be able to interface DHT sensor with Raspberry Pi and detect temperature and humidity with it.

FAQs:

- What is DHT sensor? How does it work?
- How is it interfaced with Raspberry pi/Arduino?

ASSIGNMENT NUMBER: 3

Revised On: 18-07-2022

TITLE	Write an application to capture and store the image.
PROBLEM STATEMENT /DEFINITION	Understanding and connectivity of Raspberry-Pi /Beagle board with camera. Write an application to capture and store the image.
OBJECTIVE	To capture and store image using Raspberry-pi.
S/W PACKAGESAND HARDWARE APPARATUS USED	Picamera packageRaspberry pi Camera module
REFERENCES	http://www.electronicwings.com/raspberry-pi/pi-camera-module-interface-with-raspberry-pi-using-python
STEPS	Refer to details
INSTRUCTIONSFOR WRITINGJOURNAL	Title Problem Definition Objectives Theory Class Diagram/ER diagram Test cases Program Listing Output Conclusion

Aim:
Study of Connectivity and configuration of Raspberry-Pi /Beagle board circuit with basic peripherals, LEDS. Understanding GPIO and its use in program.

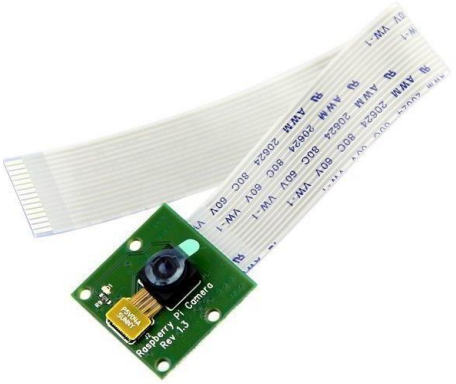
Pre-requisite:
Basic knowledge of configuration.

Learning Objectives:
To understand configuration of Raspberry-pi/Beagle board circuit with basicperipherals and its use in the program.

Learning Outcomes:
The students will be able to
Connectivity of Raspberry-pi and Implement the program

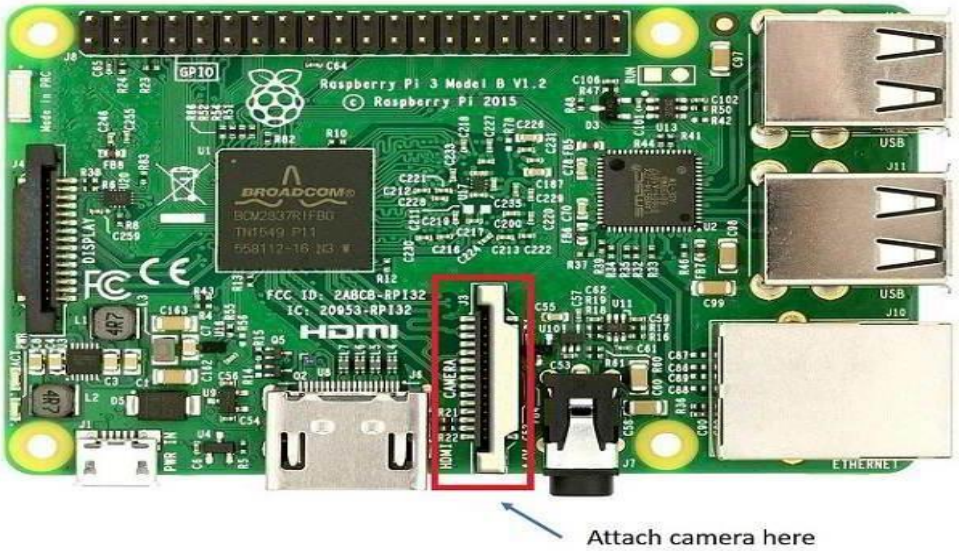
Theory:
Pi Camera Module Interface with Raspberry Pi using Python

Introduction
Pi Camera module is a camera which can be used to take pictures and high definition video.Raspberry Pi Board has CSI (Camera Serial Interface) interface to which we can attach PiCamera module directly.This Pi Camera module can attach to the Raspberry Pi‟s CSI port using 15-pin ribbon cable.



How to attach Pi Camera to Raspberry Pi?

Connect Pi Camera to CSI interface of Raspberry Pi board as shown below,



Now, we can use Pi Camera for capturing images and videos using Raspberry Pi.
Python Program for Image Capture & Storeimport picamera from time import sleep

#create object for PiCamera classcamera = picamera.PiCamera() #set resolution
camera.resolution = (1024, 768)
camera.brightness = 60

```
camera.start_preview()#add text on image
camera.annotate_text = 'Hi Pi User'sleep(5)
#store image camera.capture('image1.jpeg')camera.stop_preview()
```

Conclusion:

Students will be able to interface pi camera module with Raspberry Pi and capture image and record video using pi-camera.

FAQs:

- How do you interface camera module with Raspberry pi/Arduino?
- Which commands are used to capture image and record video?

ASSIGNMENT NUMBER: 4

Revised On: 18-07-2022

TITLE	Create a small dashboard application to be deployed on cloud.
PROBLEM STATEMENT /DEFINITION	Create a small dashboard application to be deployed on cloud. Different publisher devices can publish their information and interested application can subscribe.
OBJECTIVE	To develop comprehensive approach towards building small lowcost embedded IoT system. To understand different sensory inputs.
S/W PACKAGESAND HARDWARE APPARATUS USED	Cloud (ThingSpeak), client server model, controller/processor, Python, PC with the configuration as Latest Version of 64 bit Operating Systems, Open Source Fedora-GHz. 8 G.B. RAM, 500 G.B. HDD, 15"Color Monitor, Keyboard, Mouse
REFERENCES	Olivier Hersent, David Boswarthick, Omar Elloumi , "The Internet of Things – Key applications and Protocols", Wiley, 2012, ISBN:978-1-119-99435-0 Barrie Sosinsky, "Cloud Computing Bible", Wiley-India, 2010.ISBN : 978-0-470-90356-8 Adrian McEwen, Hakim Cassimally, "Designing the Internet of Things", Wiley, 2014, ISBN: 978-1-118-43063-7
STEPS	Refer to details
INSTRUCTIONSFOR WRITINGJOURNAL	Title Problem Definition Objectives Theory Class Diagram/ER diagram Test cases Program Listing Output Conclusion

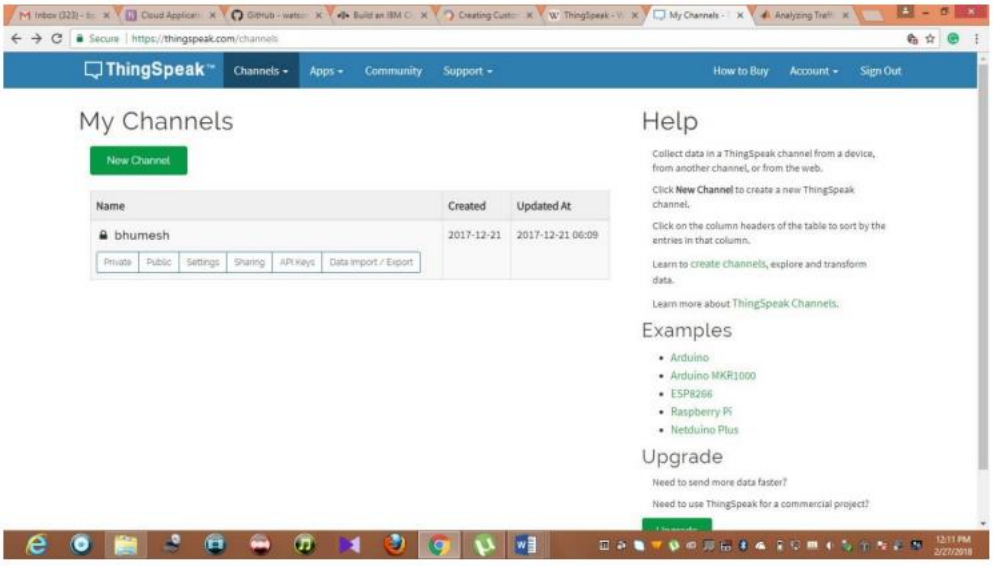
Aim: Create a small dashboard application to be deployed on cloud. Different publisher devicescan publish their information and interested application can subscribe.

Pre-requisite:
Basic knowledge of Embedded system and IOT

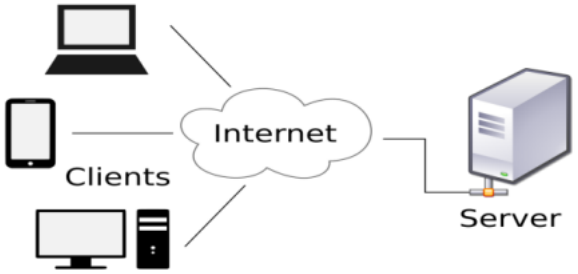
Learning Objectives:
To Develop application based on cloud.
To understand different sensory inputs
Understand client server model programming.

Learning Outcomes:
The students will be able to
Perform the connectivity with Raspberry-Pi, Beagle board, Arduino and other microcontroller.
Implement cloud application with the help of client server programming by using python.

Theory:
Thingspeak is an open-source Internet of Things (IoT) application and API to store and retrieve data from things using the HTTP protocol over the Internet or via a Local Area Network. ThingSpeak enables the creation of sensor logging applications, location tracking applications, and a social network of things with status updates. ThingSpeak was originally launched by ioBridge in 2010 as a service in support of IoT applications. For accessing Thingspeak Need to create account on https://thingspeak.com/login



Client-Server model: The client–server model is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients. Often clients and servers communicate over a computer network on separate hardware, but both client and server may reside in the same system.
A server host runs one or more server programs which share their resources with clients. A client does not share any of its resources but requests a server's content or service function. Clients therefore initiate communication sessions with servers which await incoming requests. Examples of computer applications that use the client–server model are Email, network printing, and the World Wide Web.



SETTING UP AN APACHE WEB SERVER ON A RASPBERRY PI:
Apache is a popular web server application you can install on the Raspberry Pi to allow it to serve web pages
On its own, Apache can serve HTML files over HTTP, and with additional modules can serve dynamic web pages using scripting languages such as PHP.
INSTALL APACHE
First install the apache2 package by typing the following command in to the Terminal:
sudo apt-get install apache2 -y
TEST THE WEB SERVER
By default, Apache puts a test HTML file in the web folder. This default web page is served when you browse to http://localhost/ on the Pi itself, or http://192.168.1.10 (whatever the Pi's IP address is) from another computer on the network.
Navigate to this directory in the Terminal and have a look at what's inside:
cd /var/www/html ls -al

For Developing your own application install language as preferred.
For Installing php in your server
Type the following command to install these:
sudo apt-get install php5 libapache2-mod-php5 -y
On Apache server when localhost is called by default index page get displayed. Check for version of php installed in your server with help of terminal.
Algorithm:
Install Apache server to your system(windows/linux).
Test for localhost whether page is opening as per requirement or not. (e.g., http://localhost/)
Install latest version of php in server.
Test the version and check for index.php file for proper execution.
Write a program for creating your own dashboard.
Make use of essential library functions for sending email or receiving email notifications.
Deploy the application on cloud server and check for the functionality.
WHAT THE COLUMNS MEAN
The permissions of the file or directory
The number of files in the directory (or `1` if it's a file).
The user which owns the file or directory
The group which owns the file or directory
The file size
The last modification date & time

As you can see, by default the `html` directory and `index.html` file are both owned by the `root` user.
In order to edit the file, you must gain `root`permissions. Change the owner to your own userwith `sudo chown pi: index.html` before editing.

Try editing this file and refreshing the browser to see the web page change.

YOUR OWN WEBSITE

If you know HTML you can put your own HTML files and other assets in this directory andserve them as a website on your local network.

ADDITIONAL - INSTALL PHP

To allow your Apache server to process PHP files, you'll need to install PHP5 and the PHP5module for Apache. Type the following command to install these:

sudo apt-get install php5 libapache2-mod-php5 -y

Now remove the index.html file:sudo rm index.html
and create the file index.php:sudo leafpad index.php
Note: Leafpad is a graphical editor. Alternatively, use nano *if you're restricted to the commandline*

Put some PHP content in it:

<?php echo "hello world"; ?>
Now save and refresh your browser. You should see "hello world". This is not dynamic but stillserved by PHP. Try something dynamic:

<?php echo date('Y-m-d H:i:s'); ?>or show your PHP info:
<?php phpinfo(); ?>

sudo apt-get install apache2 –y
sudo apt-get install php5 libapache2-mod-php5 -ysudo apt-get install git-core

git clone git://git.drogon.net/wiringPicd wiringPi
./build

.....now open terminal and enter this:.....

$cd /var/www
$sudo nano rahul.php

----------this will open a empty black screen window where we have to write theseinstructions----------
****
<html>
<head>
<meta name="viewport" content="width=device-width" />
<title>LED Control</title>
</head>
<body>
WEB PAGE ON PHP BASED GPIO Control:
<form method="get" action="gpio.php">
<input type="submit" value="ON" name="on">
<input type="submit" value="OFF" name="off">
</form>

```
<?php
$setmode17 = shell_exec("/usr/local/bin/gpio -g mode 17 out");if(isset($_GET['on'])){
$gpio_on = shell_exec("/usr/local/bin/gpio -g write 17 1");echo "LED is on";
}
else if(isset($_GET['off'])){
$gpio_off = shell_exec("/usr/local/bin/gpio -g write 17 0");echo "LED is off";
}
?>
</body>
</html>
-*-*-*-*-*-*-*-**-*-*-**-*__*-*-*-*-*-*-*-*-*-*
```

--------or can try this also----------

```
<?php
$myfile = fopen("/home/pi/log.txt", "r") or die("Unable to open file!");echo
fread($myfile,filesize("/home/pi/log.txt"));
fclose($myfile);
?>
```

****

-------now press "cnt+O" to save and then "cnt+X" to exit

--------once go to /var/www library there we can find rahul.php open it and cross check it---------

--------now check ip address of our pi by giving ifconfig in terminal window----------

--------enter that ip in your mobile brouser as 192.168.2.26/rahul.php------

.....bingo here it is..

Mail code: -

```
import RPi.GPIO as GPIO from subprocess import callimport time
import os import glob import smtplibimport base64
from email.mime.image import MIMEImage
from email.mime.multipart import MIMEMultipartimport subprocess
gmail_user = "checking999mail@gmail.com"gmail_pwd = "mail999checking"
FROM = 'checking999mail@gmail.com'
TO = ['hyd.embedded@pantechmail.com'] #must be a list


i=1


while (i):
i=i-1
subprocess.Popen( "fswebcam -r 1280x720 /home/pi/Downloads/pan.jpg", shell=True )
time.sleep(1)
msg = MIMEMultipart()time.sleep(1)
msg['Subject'] ="testing msg send from python"time.sleep(1)
fp = open("/home/pi/Downloads/pan.jpg", 'rb')time.sleep(1)
img = MIMEImage(fp.read())time.sleep(1)
```

fp.close() time.sleep(1)
msg.attach(img) time.sleep(1)
try:
server = smtplib.SMTP("smtp.gmail.com", 587) #or port 465 doesn't seem to work!print "smtp.gmail"
server.ehlo()
print "ehlo"server.starttls()
print "starttls" server.login(gmail_user, gmail_pwd)
print "reading mail & password" server.sendmail(FROM, TO, msg.as_string())
print "from"server.close()
print 'successfully sent the mail'
except:
print "failed to send mail"sudo apt-get install apache2
sudo apt-get install php5 libapache2-mod-php5sudo apt-get install git-core
git clone git://git.drogon.net/wiringPicd wiringPi
./build
.....now open terminal and enter this:.....
$cd /var/www/html
$sudo leafpad gpio.php
----------this will open a empty black screen window where we have to write theseinstructions---------
---
****
****
<html>
<head>
<meta name="viewport" content="width=device-width" />
<title>LED Control</title>
</head>
<body>
WEB PAGE ON PHP BASED GPIO Control:
<form method="get" action="gpio2.php">
<input type="submit" value="ON" name="on">
<input type="submit" value="OFF" name="off">
</form>
<?php
$setmode17 = shell_exec("/usr/local/bin/gpio -g mode 17 out");if(isset($_GET['on'])){
$gpio_on = shell_exec("/usr/local/bin/gpio -g write 17 1");echo "LED is on";
}
else if(isset($_GET['off'])){
$gpio_off = shell_exec("/usr/local/bin/gpio -g write 17 0");echo "LED is off";
}

?>
</body>
</html>

-*-*-*-*-*-*-*-**-*-*-**-*__-*-*-*-*-*-*-*-*-*-*

--------or can try this also----------
<?php
$myfile = fopen("/home/pi/log.txt", "r") or die("Unable to open file!");echo

fread($myfile,filesize("/home/pi/log.txt"));
fclose($myfile);
?>
****
-------now press "cnt+s" to save and then to exit
--------once go to /var/www/html library there we can find gpio.php open it and cross check it----
--------now check ip address of our pi by giving ifconfig in terminal window----------
--------enter that ip in your mobile browser as 192.168.2.26/gpio.php------
.....bingo here it is..

import subprocess
from subprocess import callimport RPi.GPIO as GPIO import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(6, GPIO.IN, pull_up_down = GPIO.PUD_UP) GPIO.setup(13, GPIO.IN ,pull_up_down = GPIO.PUD_UP)GPIO.setup(19, GPIO.IN, pull_up_down = GPIO.PUD_UP)GPIO.setup(26, GPIO.IN, pull_up_down = GPIO.PUD_UP)
while True:

if(GPIO.input(26) == 0): pageToOpen="en.wikipedia.org/wiki/A" subprocess.Popen(["midori","-a", pageToOpen]) call(["espeak","Object found is an apple"],shell=True)time.sleep(5)
else:
print('a')
time.sleep(0.5)
#GPIO.cleanup()

Conclusion:
Thus, students will be able to install and configure the server and the programming language. Also, students will be able to host their application with subscription and subscription option.

Review Questions:
1.Explain concept pf server in detail.
2.Explain significance of client server architecture with its use.
3.What are the php tags used for sending emails? Explain them with proper example.
4.What is cloud? How to store data on cloud?
5.What is an embedded system? How is it used on cloud?

PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE

ACADEMIC YEAR: 2023-24

DEPARTMENT of COMPUTER ENGINEERING DEPARTMENT

CLASS: T.E. **SEMESTER: I**

SUBJECT: LP-1 (HCI)

ASSINGMENT NO.	1
TITLE	Design a paper prototype for selected Graphical User Interface.
PROBLEM STATEMENT /DEFINITION	Develop ideas and designing user flows using hand-sketched screens that represent a digital product for any application.
OBJECTIVE	To process where design teams create paper representations of digital products to help them realize concepts and test designs
OUTCOME	To intend for a definite purpose, to assign in thought or intention**.**
S/W PACKAGES AND HARDWARE/ APPARATUS USED	Apparatus: Paper and pen
REFERENCES	https://www.uxpin.com/studio/blog/paper-prototyping-the-practical-beginners-guide/
STEPS	1. Use printer paper and cheap pencils/pens. Ruled or line pads often stifle creativity as designers get side-tracked drawing between the lines rather than developing lots of ideas. 2. Start with a warm-up! Sometimes it takes a few sketches to loosen up and get into the flow. Crazy eights is a fantastic paper prototyping method to design many versions of the same screen fast. After a couple of crazy eight rounds, you'll have many ideas to expand on. 3. Prototype mobile-first or progressive enhancement starts with the smallest screen and adjusts the layout as you scale the viewport (this applies to mobile and web design. Scaling up is much easier than scaling down because you prioritize content and avoid elaborate desktop layouts that don't translate to mobile. 4. Stick to one sketch per screen (a piece of paper). Paper prototyping requires you to create user flows by placing pieces of paper in sequences. You'll also switch these around or add new screens. If you have more than one screen on a piece of paper, you lose this speed and flexibility. 5. Iterate as the ideas come to mind. The goal is quantity, not quality. When you create lots of paper prototype ideas, you often end up taking bits from each to get the final result—like a Lego set, but with paper.
INSTRUCTIONS FOR WRITING JOURNAL	1. Date 2. Assignment no.

	3. Problem definition 4. Learning objective 5. Learning Outcome 6. Concepts related Theory 7. Algorithm 8. Test cases 9. Conclusion/Analysis

Prerequisites: UI design and Software design

Concepts related Theory:

With tools like reMarkable and Apple Pencil, teams can collaborate remotely while enjoying the speed and versatility of the physical paper experience. Using digital sketch tools can accelerate the paper prototyping process. Designers can make changes faster (without needing to redraw a screen), attach detailed notes, and upload finished prototypes instantly to design tools like UXPin to build high-fidelity prototypes or go with wireframing. Paper prototyping digitally also reduces paper and plastic waste, which is better for the environment.

Paper prototyping is the fun part of product design. It's an opportunity for team members to brainstorm and sketch ideas. Don't worry about how beautiful your sketches look. Even the best UX designers aren't brilliant sketch artists! The goal is to visualize your ideas and get the creative juices flowing.

Algorithm: (Testing)

Testing paper prototypes:

- Designate one person other than the presenter as play the "human computer" or product simulator – The person playing the human-computer will simulate scrolling, swiping, navigating to different screens, and other functionality.
- Rehearse – Rehearsing is essential so that the presenter and simulator are in sync. The presenter can work out a good cadence for the simulator to keep up with the presentation.
- Follow standard usability test best practices – Standards like using a minimum of 5 users and recording the tests still apply. You can download our free Guide to Usability Testing for more understanding of usability standards and practices.
- If you're giving users a paper prototype to inspect, ensure you provide guidance and annotations, so they know where to focus and what they're supposed to test.

Conclusion:

Paper prototyping can be extremely helpful during the early-stage conceptualizing — when a team needs to explore a variety of different concepts and choose the one that will be used. The paper

prototype allows quickly visualize and test various ideas.

Review Questions:

1) How do you make a paper prototype?
2) What is a prototype in user interface design?
3) What are the features of paper prototype?
4) How prototype is useful in GUI?
5) How do you create a prototype in design thinking?

ASSINGMENT NO.	2

TITLE	Implement GOMS (Goals, Operators, Methods and Selection rules) modeling technique to model user's behavior in given scenario
PROBLEM STATEMENT /DEFINITION	Implement GOMS (Goals, Operators, Methods and Selection rules) modeling technique to model user's behavior in given scenario
OBJECTIVE	To improve the efficiency of human-machine interaction by identifying and eliminating unnecessary user actions.
OUTCOME	*Goals*, *operators*, *methods*, and *selection rules* is a *method* derived from human-computer interaction (HCI) and constructs a description of human performance.
S/W PACKAGES AND HARDWARE/ APPARATUS USED	Critical-Path Model GOMS (**CPM-GOMS**) Natural GOMS language (**NGOMSL**)
REFERENCES	http://cs4760.csl.mtu.edu/2016/lectures/goals-operators-methods-selection-goms/
STEPS	1) Goals: A goal is something that the user tries to accomplish. The analyst attempts to identify and represent the goals that typical users will have. A set of goals usually will have a hierarchical arrangement in which accomplishing a goal may require first accomplishing one or more subgoals. 2) Operators are actions that the user executes. There is an important difference between goals and operators. Both take an action-object form, such as the goal of revising document and the operator of Keystroke ENTER. But in a GOMS model, a goal is something to be accomplished, while an operator is just executed. This distinction is intuitively based, and is also relative; it depends on the level of analysis chosen by the analyst (John & Kieras, 1996) 3) Methods: A method is a sequence of steps that accomplishes a goal. A step in a method typically consists of an external operator, such as pressing a key, or a set of mental operators involved with setting up and accomplishing a subgoal. Much of the work in analyzing a user interface consists of specifying the actual steps that users carry out in order to accomplish goals, so describing the methods is the focus of the analysis. 4) Steps More than one operator can appear in a step, and at least one step in a method must contain the operator Return_with_goal_accomplished. A step starts with the keyword Step, contains an optional label, followed by a period, and one or more operators separated by semicolons, with a final period:
INSTRUCTIONS FOR WRITING JOURNAL	1. Date 2. Assignment no. 3. Problem definition 4. Learning objective 5. Learning Outcome 6. Concepts related Theory 7. Algorithm 8. Test cases 9. Conclusion/Analysis

Prerequisites: Software Engineering & Design

Concepts related Theory:

Keystroke-level Model GOMS (KLM-GOMS)

Card, Moran, and Newell (The Keystroke-level Model for User Performance with Interactive Systems, Communications of the ACM, 23:396-410, 1980) measured the time for users to perform a series of gestures on the computer. They discovered a fundamental principle:

The total time to perform a sequence of gestures is the sum on the individual gestures.

A lot is implied in this statement. The most important is that there are fundamental gestures. Individual users perform the fundamental gestures in different times; the researchers attempted to determine typical values:

K = 0.2 sec Keying: The time to perform a keystroke, or mouse click

P = 1.1 sec Pointing: The time to position the mouse pointer

H = 0.4 sec Homing: The time for user to move hands from keyboard to mouse

M = 1.35 sec **Mental**: The time for the user to prepare for the next step

R =? **Responding**: The time for the computer to respond to the user inputs.

The variation of the timings across users can be as much as 100%, for example an expert typist can type 200 words per minute = 0.06 sec (Note that the measurement assumes 5 characters/words). So the model cannot accurately predicate the response time of an individual user. Chris Blazek and I have measured these variables for a web user and they are surprisingly accurate. Even without precise gesture times for a specific user, the model can be used to determine times for expert users and compare across interfaces.

We calculate the total response time by listing the individual gesture and summing their individual execution time. The difficult part is determining where a mental preparation, **M**, occurs. The researchers determined heuristics rules for placing mental operations:

Rule 0: Initial insertion of candidate Ms: Insert M before all Ks and Ps

Rule 1: Deletion of anticipated Ms: If P or K is fully anticipated by a preceding P or K then delete the middle M. For example, moving the mouse to tap on the button; PMK => PK

Rule 2: Deletion of Ms in cognitive units: If a series of Ks represent a string, then delete the middle Ks; for example, type '1.2' is a cognitive unit; MKMKMK => MKKK

Rule 3: Deletion of Ms before consecutive terminators: If several delimiters are typed only keep the first M. For example, if '))' is the terminator, use only one M.

Rule 4: Deletion of Ms that are terminators of commands: If the terminator is a frequently used, delete the M before the terminator; for example, a command followed by "return," so the M before the K representing the "return" is deleted. But if the terminator delimits arguments for a command string that vary then keep the M. This represents checking that the arguments are correct.

Rule 5: Deletion of overlapped Ms: Do not count any portion of an M that overlaps with a command response. (This is the reason that a responsive interface only needs to respond in a second.)

Algorithm:

We want to design an interface to convert centimetres to inches and vice a versa.

This is a simple problem, but we develop 4 designs.

Design 1: Dialog box

Interface consists of two radial buttons to choose between centimetres or inches and two text fields; one text field to type the 4 characters for distance and the other to display the result.

Sequence of Tasks:

1. Move hand to mouse,
2. Move mouse to units' radio button,
3. Click on units,
4. Move hand to keyboard,
5. Type 3 characters,
6. Type enter.

Design 2: A GUI

Sequence of gestures:

1. Move hand to mouse.
2. Move mouse to pointer.
3. Click mouse.
4. Move pointer, dragging the pointer to a new distance.
5. Release pointer

Design 3: Message box request for information

A message box appears asking the user to input the units (cm/inches), the distance (4 digits) followed by return/enter. Then the keystroke sequence is MK KKKK MK = 3.9 sec.

Design 4: Bifurcated output

A box appears with text field for entering 3 digits and automatically there appears two output text fields, one text field conversion in centimetre the other in inches.

The required keystrokes are MKKKK = 2.15 sec. *IDEAL*.

Conclusion:

GOMS is a specialized human information processor model for human-computer interaction observation that describes a user's cognitive structure on four components

Review Questions:

1) What are operators in GOMS model?
2) How do you do GOMS analysis?
3) What is the mental operator in KLM?
4) How GOMS are used in technical documentation?
5) What is KLM and GOMS?

ASSIGNMENT NO.	3
TITLE	Design a User Interface in Python.

PROBLEM STATEMENT /DEFINITION	Design a User Interface in Python.
OBJECTIVE	To improve the efficiency of human-machine interaction by identifying and eliminating unnecessary user actions.
OUTCOME	Build a Python GUI Application With wxPython
S/W PACKAGES AND HARDWARE/ APPARATUS USED	Python and IDLE, python package wxpython
REFERENCES	1. https://realpython.com/python-gui-with-wxpython/#installing-wxpython 2. https://www.uxmatters.com/mt/archives/2009/02/reviewing-user-interfaces.php
STEPS	**1) Installing wxPython** User interfaces have some common components: ● Main window ● Menu ● Toolbar ● Buttons ● Text Entry ● Labels 2) **Creating a Skeleton Application** ● Widgets ● Absolute Positioning ● Sizers (Dynamic Sizing) ● Adding an Event 3) **Creating a Working Application** For example - ● mp3-tagger ● eyeD3 ● mutagen 4) **Creating the User Interface** 5) **Make a Functioning Application** ● Creating an Editing Dialog
INSTRUCTIONS FOR WRITING JOURNAL	1. Date 2. Assignment no. 3. Problem definition 4. Learning objective 5. Learning Outcome 6. Concepts related Theory 7. Algorithm 8. Test cases 9. Conclusion/Analysis

Prerequisites: Software Engineering & Design

Concepts related Theory:

There are many graphical user interface (GUI) toolkits that you can use with the Python programming language. The big three are Tkinter, wxPython, and PyQt. Each of these toolkits will work with Windows, macOS, and