1.

#include<unistd.h>

#include<sys/types.h>

#include<stdio.h>

main (){

pid\_t pid;

pid=fork();

if (pid>0) {

printf("hello");

} else if (pid==0) {

printf("Have you heard about % d", pid );

}

return 0;

}

2.

#include <stdlib.h>

#include <stdio.h>

#include <unistd.h>

#include <errno.h>

#include <sys/wait.h>

#include <sys/types.h>

int main(void) {

pid1=fork();

pid2=fork();

pid3=fork();

pid4=fork();

pid5=fork();

pid6=fork();

pid7=fork();

if (pid1 > 0) {

waitpid(pid1, NULL, 0);

}

if (pid2 > 0) {

waitpid(pid2, NULL, 0);

}

if (pid3 > 0) {

waitpid(pid3, NULL, 0);

}

if (pid4 > 0) {

waitpid(pid4, NULL, 0);

}

if (pid5 > 0) {

waitpid(pid5, NULL, 0);

}

if (pid6 > 0) {

waitpid(pid6, NULL, 0);

}

if (pid7 > 0) {

waitpid(pid7, NULL, 0);

}

return 0;

}

3.

#include <iosteram>

#include <stdio.h>

int main(int argc, char\*\* argv) {

int n=atoi(argv[1]);

std::cout<< "hello "<<n<< std::endl;

return 0;

}

parent.cpp: …

# include <windows.h>

Int main() {

STARTUPINFO si;

PROCESS\_INFORMATION pi;

si.cb = sizeof(si);

ZeroMemory(&si, sizeof(STARTUPINFO));

ZeroMemory(&pi, sizeof(PROCESS\_INFORMATION));

CreateProcessA(NULL, (LPSTR) (command + " " + std::to\_string(i)).c\_str(),

NULL, NULL, FALSE, CREATE\_NEW\_CONSOLE | CREATE\_SUSPENDED, NULL, NULL,

&si, &pi);

WaitForMultipleObjects(thread\_handle, INFINITE);

std::cout<< "and"<< std::endl;

….

return 0;

}

4.

#include <stdlib.h>

#include <stdio.h>

#include <unistd.h>

#include <errno.h>

#include <sys/wait.h>

#include <sys/types.h>

int main(void) {

return 0;

}