ZAŁĄCZNIK NR 1

clone.c

```
#include<stdlib.h>
#include<stdio.h>
#include<unistd.h>

#include <sys/types.h>
#include <sys/wait.h>
#include <sched.h>
#include <linux/sched.h>

int zmienna_globalna=0;

#define ROZMIAR_STOSU 1024*64

int funkcja_watku( void* argument )
{

  zmienna_globalna++;
  printf(" ");
  /* int wynik; */
  /* wynik=execv("./program",NULL); */
  /* if(wynik==-1) */
  /*   printf("Proces potomny nie wykonal programu\n");
*/

  return 0;
}

main()
{

  void *stos;
  pid_t pid;
  int i;

  stos = malloc( ROZMIAR_STOSU );
  if (stos == 0) {
    printf("Proces nadrzędny - blad alokacji stosu\n");
    exit( 1 );
  }
  inicjuj_czas();
  for(i=0;i<1000;i++){

    pid = clone( &funkcja_watku, (void *)
stos+ROZMIAR_STOSU,
                        CLONE_FS | CLONE_FILES |
CLONE_SIGHAND | CLONE_VM, 0 );

    waitpid(pid, NULL, __WCLONE);

  }
  drukuj_czas();
  free( stos );
}
```

fork.c

```
#include<stdlib.h>
#include<stdio.h>
#include<unistd.h>

int zmienna_globalna=0;

main(){

  int pid, wynik, i;
  inicjuj_czas();
  for(i=0;i<1000;i++){

    pid = fork();

    if(pid==0){

      zmienna_globalna++;
      printf(" ");
      /* wynik=execv("./program",NULL); */
      /* if(wynik==-1) */
      /*     printf("Proces potomny nie wykonal
programu\n"); */

      exit(0);

    } else {

      wait(NULL);

    }
  }
  drukuj_czas();
}
```

ZAŁĄCZNIK NR 2

program.c

```
#include<stdlib.h>
#include<stdio.h>
#include<unistd.h>

int main(){

    printf("Marcin Lichota");
    return 0;

}
```

clone.c

```
#include<stdlib.h>
#include<stdio.h>
#include<unistd.h>

#include <sys/types.h>
#include <sys/wait.h>
#include <sched.h>
#include <linux/sched.h>

int zmienna_globalna=0;

#define ROZMIAR_STOSU 1024*64

int funkcja_watku( void* argument )
{

  zmienna_globalna++;
  int wynik;
  wynik=execv("./program",NULL);
  if(wynik==-1)
     printf("Proces potomny nie wykonal
programu\n");

  return 0;
}

main()
{

  void *stos;
  pid_t pid;
  int i;

  stos = malloc( ROZMIAR_STOSU );
  if (stos == 0) {
    printf("Proces nadrzędny - blad alokacji
stosu\n");
    exit( 1 );
  }
  inicjuj_czas();
  for(i=0;i<1000;i++){

    pid = clone( &funkcja_watku, (void *)
stos+ROZMIAR_STOSU,
                      CLONE_FS |
CLONE_FILES | CLONE_SIGHAND |
CLONE_VM, 0 );

    waitpid(pid, NULL, __WCLONE);

  }
  drukuj_czas();
  free( stos );
}
```

fork.c

```
#include<stdlib.h>
#include<stdio.h>
#include<unistd.h>

int zmienna_globalna=0;

main(){

  int pid, wynik, i;
  inicjuj_czas();
  for(i=0;i<1000;i++){

    pid = fork();

    if(pid==0){

      zmienna_globalna++;
      wynik=execv("./program",NULL);
      if(wynik==-1)
          printf("Proces potomny nie
wykonal programu\n");

      exit(0);

    } else {

      wait(NULL);

    }
  }
  drukuj_czas();
}
```

ZAŁĄCZNIK NR 3

```
#include<stdlib.h>
#include<stdio.h>
#include<unistd.h>

#include <sys/types.h>
#include <sys/wait.h>
#include <sched.h>
#include <linux/sched.h>

int zmienna_globalna=0;

#define ROZMIAR_STOSU 1024*64

int funkcja_watku( void* argument )
{
  int zmienna_lokalna = 0;
  for(int i = 0; i < 1000000; i++) {
      zmienna_globalna++;
      zmienna_lokalna++;
  }
  printf("Zminna_lokalna: %d\n",zmienna_lokalna);
  printf("Zminna_globalna: %d\n",zmienna_globalna);
  return 0;
}

main()
{

  void *stos;
  void *stos2;
  pid_t pid, pid2;
  int i;

  stos = malloc( ROZMIAR_STOSU );
  stos2 = malloc( ROZMIAR_STOSU );
  if (stos == 0) {
    printf("Proces nadrzędny - blad alokacji stosu\n");
    exit( 1 );
  }
  if (stos2 == 0) {
    printf("Proces nadrzędny - blad alokacji stosu2\n");
    exit( 1 );
  }
  pid = clone( &funkcja_watku, (void *) stos+ROZMIAR_STOSU,
   CLONE_FS | CLONE_FILES | CLONE_SIGHAND | CLONE_VM, 0 );
  pid2 = clone( &funkcja_watku, (void *) stos2+ROZMIAR_STOSU,
    CLONE_FS | CLONE_FILES | CLONE_SIGHAND | CLONE_VM, 0 );

  waitpid(pid, NULL, __WCLONE);
  waitpid(pid2, NULL, __WCLONE);

  free( stos );
  free( stos2 );
}
```